Models i mètriques d'estimació d'esforç per la gestió de projectes informàtics.

TFC: Sistemes d’informació. Gestió de projectes

Consultor: Jesús Bustínduy Basterrechea
Alumne: Francesc Tarragó Moya
Semestre: Primavera 2004

Resum

El objectiu principal del treball ha estat buscar i valorar les diferents opcions que existeixen en la gestió de projectes per estimar els costos relacionats. La primera de les tasques que s'ha fet és definir el terme de projecte per esbrinar quins tipus de projectes són susceptibles de ser valorats, aquesta tipificació ens ha portat a dividir els projectes en dues rames molt diferenciades encara que relacionades durant el seu desenvolupament. Les rames esmentades són els projectes de gestió relacionats amb la creació de programari i els que no necessiten d'una manera explicita un desenvolupament de programari. En realitat tal com s'ha comentat en moltes gestions de projectes fa falta una valoració de creació de programari encara que no sigui el nucli del mateix i a l'inrevés ens trobem amb l'estimació de certs costos de recursos utilitzats no relacionat amb la programació. Per arribar la tipificació s'han definit conceptes generals com les característiques, les etapes, els recursos i els costos relacionats en els diferents projectes

El següent pas ha estat la troballa de les mètriques aplicables per la valoració, seguin l'estructura del primer bloc s'ha definit mètrica dins de la gestió i s'han buscat i valorat les diferents opcions depenent dels recursos utilitzats, hem pogut veure que les mètriques per la valoració de projectes de enginyeria de programari són en l'actualitat més acotades i a nivell de gestió molt més definides.

L'últim bloc tracta els models que existeixen, models que utilitzen les mètriques per les valoracions. En aquest bloc podem trobar un grup de models força documentats això si, són models pensat per les mètriques de creació de programari i utilitzats per la gestió de projectes d'enginyeria de programari, si mes no, no hauria de ser una sorpresa si valorem l'historia de la informàtica des de els seus inicis, on la importància del desenvolupament ha anat perdent força per equiparar-se a la resta de tasques dins de l'àmbit informàtic.

Índex

Resum 2

Índex 3

Cos de la memòria 5

1.-Introducció 5

1.1.- Justificació del TFC i context :punt de partida i aportació del TFC. 5
1.2.- Objectius del TFC 6
1.3.- Enfocament i mètode seguit 6
1.4.- Planificació del projecte 7
1.4.1.-Establiment de calendari 7
1.4.2.-Descomposició de tasques (WBS) 7
1.4.3.-Diagrama de Gantt 7
1.4.4.-Fites de control 8
1.5.- Productes obtinguts 8
1.6.- Breu descripció dels altres capítols 8

2.- Els projectes informàtics 9

2.1.- Característiques diferencials 10
2.1.1.- És concret 10
2.1.2.- És excepcional 10
2.1.3.- Té una durada limitada 11
2.1.4.- És flexible. 11
2.2.- Les etapes d'un projecte i el cicle de la vida 11
2.2.1.- L'inici del projecte 11
2.2.2.- La qualificació del projecte. 11
2.2.3.- El desenvolupament del projecte. 12
2.2.4.- El tancament del projecte. 12
2.2.5.- El cicle de la vida 12
2.4.- Els recursos i els costos reals 13
2.4.1.- El maquinari de les màquines objectiu. 13
2.4.2.- El maquinari de proves i desenvolupament. 13
2.4.3.- El programari utilitzat. 13
2.4.4.- Els recursos humans. 13
2.4.5.- Els costos reals 14
2.5.- Tipus de projectes una visió empresarial. 15
2.5.1.- Projectes informàtics no relacionats amb la creació de programari. 15
2.5.1.1.- Projectes referents a la gestió de servidors. 15
2.5.1.2.- Projectes referents a la gestió d'usuaris. 16
2.5.1.3.- Projectes referents a la gestió de xarxes. 16
2.5.1.4.- Projectes referents a la gestió de les dades. 16
2.5.1.5.- Projectes referents a la gestió de l'entorn WEB. 16
2.5.1.6.- Projectes referents a la gestió de la seguretat 16
2.5.2.- Projectes informàtics de creació de programari. 16

3.- Abast actual, mètriques i models d'estimació d'esforç. 18

3.1.- Estimació de costos en la gestió d'un projecte informàtic 18
3.1.1.- Moment de l'estimació i grau d'exactitud 19
3.2.-Mètriques per la gestió d'un projecte informàtic 20
3.2.1.-Mètriques per la construcció de programari 21
3.2.1.1.- Mètriques del producte i del procés 21
3.2.1.2.- Mètriques de qualitat i de productivitat 21
3.2.1.3.- L'esforç i la mesura de la productivitat 22
3.2.1.4.- Mètriques orientades a la grandària i a la funció 23
3.2.2.- Mètriques aplicables a la gestió de projectes informàtics no de construcció de programari. 30

3.3.- Models per la gestió d'un projecte informàtic. 31
3.3.1.- Els mètodes d'estimació possibles segons Barry W. Boehm 31
3.3.2.- Models d'estimació en la construcció de programari. 32
3.3.2.1.- Models històrics 34
3.3.2.2.- Models empírics o estadístics 34
3.3.2.3.- Models teòrics: el model de Putman 36
3.3.2.4.- Models compostos: els models COCOMO de Barry W. Boehm 38
3.3.2.5.- L'ús d'estàndards de productivitat 41
3.3.3.- Aplicació dels mètodes i models d'estimació en la gestió d'altres tipus de projectes informàtics 43

4.- Punts generals per l'estimació d'esforç en la gestió de projectes. 44

5.- Conclusions 45

6.-Glossari 46

7.-Bibliografia. 48

8.- Annexos 50

Cos de la memòria

1.-Introducció

En els estudis de la gestió de projectes informàtics es defineixen models i mètriques que ajuden en l'estudi de l'estimació de l'esforç necessari per arribar a concretar el cost final dels projectes. L'enfocament en aquest estudis es centre en l'àmbit de la creació de nou programari deixant un gran ventall de necessitats actuals .

Aquest treball intentarà omplir aquest vuit fen un treball de recerca per esbrinar si existeix tant metodologies com mètriques per els projectes informàtics en general, sense deixar de banda els empleats en la creació de programari, part àmpliament definida actualment, e intentant proposar-ne en cas de no trobar res relacionat. S'intentarà crear una guia, manual que permeti valorar les diferents opcions possibles i pugui ser un mòdul de consulta eficient per la valoració de projectes informàtics.

Per assolir aquest propòsit en serà necessari establir les diferents classes de projectes informàtics susceptibles de ser valorats i estimats tant a nivell d'esforç de recursos com de cost.

1.1.- Justificació del TFC i context :punt de partida i aportació del TFC.

L'evolució constant de la enginyeria informàtica enfront de les necessitats del mercat, marquen els requeriments cada vegada més aguditzades de una bona gestió del projectes.

En general el tema informàtica esta en un procés de canvi constant, fet que ens obliga a generalitzà força els conceptes inicials que ens permeten du a terme una gestió acurada dels diferents projectes. Aquest fet sumat a que les inversions no són tot el que nosaltres voldríem, ens porta a intentar racionalitzar qualsevol projecte de manera que ens permeti estimar i controlar els efectes de les decisions presses durant la gestió.

El TFC intenta d'una manera real trobar sistemes que ens permetin gestionar els projectes. Per fer-ho esbrinem quines són les mètriques aplicables i quins models ja definits ens poden servir de guia en part o en la seva totalitat per gestionar qualssevol projecte informàtic.

El TFC està recolzat en la documentació de l'assignatura Gestió d'una organització informàtica on s'estimen perfectament les mètriques i els models de gestió de projectes de creació de programari, part sobre la que fa un repàs minuciós. Degut a l'evolució del mon informàtic esmentat anteriorment s'espera trobar altre models i mètriques aplicables a projectes informàtics que no tinguin que veure amb l'enginyeria de programari. L'aportació esperada és una recopilació a dia d'avui de les mètriques i els models aplicables a la gestió de projectes informàtics.

1.2.- Objectius del TFC

- Definir el concepte de projecte informàtic mitjançant les característiques, etapes i recursos.
- Determinar les possibles classes de projectes informàtics .
- Esbrinar qualsevol mètrica o model aplicable a l'estimació d'esforç dels projectes informàtics.
- Estudiar les necessitats actuals a l'hora de valorar l'esforç del projecte informàtic.
- Donar una visió real de les possibilitats per validar un projecte informàtic.
- Intentar fer propostes per una millor estimació de l'esforç dels projectes informàtics.

1.3.- Enfocament i mètode seguit

El objectiu que s'ha buscat sempre és definir les mètriques i els models existents en l'actualitat per valorar projectes informàtics.

El primer punt ha estat definir els diferents projectes sobre els que s'hauran de valorar les mètriques i els models. Per assolir o catalogar la diversitat de projectes informàtics, és interessant fer una diferencia entre els projectes purament acadèmics, dels quals és menciona una tipificació, i els projectes empresarials, es a dir tots aquells projectes que en la realitat diària de treball són susceptibles de ser gestionats . En determinats moments s'han valorat altre tipus de projectes informàtics relacionats com per exemple "outsourcing", encara que no s'ha puntualitzat degut a que el objectiu és donar una visió al lector, sense aprofundir en cada un dels projectes , per què valori quines mètriques o models pot esbrinar i aplicar al projecte. Hi ha casos on els projectes es poden dividir en subconjunts més petits per la seva gestió, punt que és determinant per aconseguir el objectiu final que és una bona gestió.

Una vegada s'han pogut tipificar d'una manera general ,però no amb la intenció de fer una tipificació perfecta de projectes, s'ha introduït el concepte mètrica per la seva posterior localització de cada projecte definit. Em de tenir present que hi ha mètriques que poden semblar estàndard com els tems però ni ha d'altres que encara que ho semblin segons el projecte poden tenir connotacions diferents com per exemple la longitud.

Si sabem definir be les mètriques associades al nostre projecte finalment ja estem en disposició de buscar si existeix algun model aplicable. Al igual que ens passava amb les mètriques hi ha models que sempre poden ser aplicats com per exemple els històrics, d'altres dependrà de les mètriques que utilitzem durant la seva definició.

1.4.- Planificació del projecte

1.4.1.-Establiment de calendari

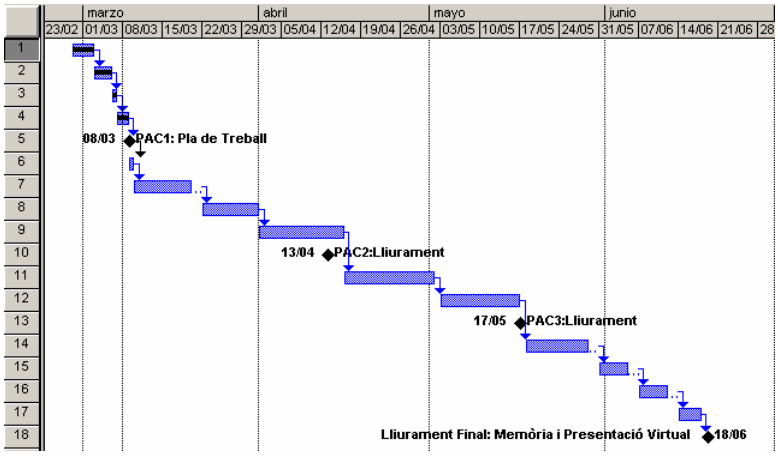
El fet d'estudiar a la UOC fa que qualssevol dia sigui útil per estudiar i treballar amb el projecte. Segons calendari només i han 3 dies de festa el 9/4 , 12/4 i 24/4 però aquest semestre els utilitzaré tots.

Les dates hàbils son del 24/02 al 18/06.

1.4.2.-Descomposició de tasques (WBS)

Descomposició de tasques			
Idn	Nom activitat	Duració (dies)	Precedències
1	Selecció tema	3	-
2	Proposta tema al consultor	3	1
3	Definició tema	1	2
4	Confecció Pla Treball	2	3
5	Confecció (*Introducció, Objectius*)	1	4
6	Confecció capítol 3,3.1,3.2 (*Els Projectes informàtics, Etapes, Característiques*)	9	4
7	Confecció capítol 3.3,3.4,3.5 (*Recursos, El cicle de la vida, Els costos reals)*	8	4
8	Confecció capítol 4,4.1 (*Abast actual, Mètriques)*	11	4
9	Confecció capítol 4.2 (A*bast actual, Models)*	12	4
10	Confecció capítol 5 (*Necessitats actuals i futures)*	11	4
11	Confecció capítol 6,7,8 (*corol·lari, glossari i bibliografia)*	9	4
12	Confecció Memòria	5	11
13	Confecció Presentació	5	12
14	Revisió	4	13
Total		**84**	

1.4.3.-Diagrama de Gantt



		Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	✓	Selecció tema	3 días	sá 28/02/04	ma 02/03/04	
2	✓	Proposta tema al consultor	3 días	mi 03/03/04	vi 05/03/04	1
3	✓	Definició tema	1 día	sá 06/03/04	sá 06/03/04	2
4	✓	Confecció Pla Treball	2 días	do 07/03/04	lu 08/03/04	3
5	✓	PAC1: Pla de Treball	0 días	lu 08/03/04	lu 08/03/04	4
6		Confecció (Introducció, Objectius)	1 día	ma 09/03/04	ma 09/03/04	5
7		Confecció capitol 3,3.1,3.2 (Els Projectes informàtics, Etapes, Característiques)	9 días	mi 10/03/04	do 21/03/04	6
8		Confecció capitol 3.3,3.4,3.5 (Recursos, El cicle de la vida, Els costos reals)	8 días	lu 22/03/04	mi 31/03/04	7
9		Confecció capitol 4,4.1 (Abast actual, Mètriques)	11 días	ju 01/04/04	ju 15/04/04	8
10		PAC2:Lliurament	0 días	ma 13/04/04	ma 13/04/04	
11		Confecció capitol 4.2 (Abast actual, Models	12 días	vi 16/04/04	sá 01/05/04	9
12		Confecció capitol 5 (Necesitats actuals i futures)	11 días	lu 03/05/04	do 16/05/04	11
13		PAC3:Lliurament	0 días	lu 17/05/04	lu 17/05/04	
14		Confecció capitol 6,7,8 (corol·lari, glossari i bibliografia)	9 días	ma 18/05/04	do 30/05/04	12
15		Confecció Memòria	5 días	lu 31/05/04	do 06/06/04	14
16		Confecció Presentació	5 días	lu 07/06/04	do 13/06/04	15
17		Revisió	4 días	lu 14/06/04	ju 17/06/04	16
18		Lliurament Final: Memòria i Presentació Virtual	0 días	vi 18/06/04	vi 18/06/04	17

1.4.4.-Fites de control

Fites de control		
Títol	**Contingut**	**Data**
Lliurament PAC1	Pla de treball	08/03/04
Lliurament PAC2	Capítols 1,2,3,4.1	13/04/04
Lliurament PAC3	Capítols 4.2,5	17/05/04
Lliurament Final	Capítols 6,7,8 Memòria i Presentació Virtual	18/06/04

1.5.- Productes obtinguts

Partint de l'objectiu inicial, definir les mètriques i els models existents en l'actualitat per valorar projectes informàtics, el resultat ha estat un document introductori per a la gestió de projectes informàtics. Inicialment es volia intentar crear o deduir possibles models no documentats, i trobar solucions de gestió per projectes futurs. La realitat és que fa falta molt més temps per poder finalitzar amb els objectius inicials. L'actual document pot ser el punt de partida que doni la informació necessària per introduir-nos en el mon de la gestió i ens permeti estructura les noves idees.

1.6.- Breu descripció dels altres capítols

La informació del TFC està estructurada amb 3 parts:

La primera, la definició de projecte i la seva possible tipificació des d'una visió empresarial, buscant el punt més real amb el treball informàtic, mencionant la opció dels projectes acadèmics i obviant tipus de projectes molt específics o més relacionats amb el mon de la enginyeria.

La segona introdueix el conceptes de mètrica, mètodes i model esbrinant tot el relacionat amb aquest conceptes fen una classificació segons les unitats de mesura i l'experiència.

Finalment busquem trobar punt comuns en la gestió de projectes.

2.- Els projectes informàtics

La definició de projecte és un concepte força genèric però molt útil si el que volem és arribar a definir el conjunt dels informàtics.
Segons el Diccionari de la llengua catalana, un projecte, en general, és 'allò que hom pensa portar a acompliment' i també, 'pla proposat per a realitzar-ho'. Un projecte és assolir quelcom i, tal com diu el diccionari, també és la realització detallada de una tasca.

Les característiques diferencials d'un projecte són que té un objectiu clar, que podem identificar un conjunt de tasques i que és necessari la intervenció d'un especialista, evidentment hi ha limitacions en els recursos però sempre hi ha un principi i un final en el temps. Requerim qualitat i sempre fa falta una planificació.

Em de tenir present que en el àmbit de la informàtica hi ha moltes coses a dur a terme i, de fet, es pot pensar en tota mena de projectes, però abans intentem esbrinar els possibles projectes que podem trobar d'una manera més general.

Segons Dawson, C.W. podem englobar un projecte a nivell acadèmic dintre de cinc categories:

- Projectes basats en la investigació.

  Aquest tipus de projectes impliquen un anàlisi amb cura d'un àrea en particular, millorant la comprensió d'aquesta. És necessari identificar els punts forts i dèbils en el camp que estem estudiant. S'han d'esbrinar, trobar els punts susceptibles de un desenvolupament posterior i especificar clarament els objectius i beneficis de la investigació sempre dins d'un marc ètic.

Requereixen una bona troballa i anàlisi bibliogràfic.

- Projectes de desenvolupament.

  Quant parlem de desenvolupament ens trobem en un àrea força amplia, degut a que podem trobar molts motius per crear un desenvolupament.
  Des de aquest punt de vista sens pot permetre dividir-los entre :
  Desenvolupament de Sistemes de software, hardware, telemàtica i desenvolupament de models de processos, mètodes i algoritmes.
  Qualsevol tipus de desenvolupament requereix una certa avaluació conegut com estudi de viabilitat i una documentació en línia, de requisits, de disseny, d'anàlisi i de proves. En alguns casos tenim que crear manuals o guies
  per els usuaris per que aquest pugui utilitzar correctament i amb el mínim esforç el nostre desenvolupament.

- Projectes d'avaluació.

  El objectiu principal és alguna forma d'anàlisi o valoració, una de les seves característiques és el estudi crític o estimació de diferents perspectives.

Dawson, C.W. and Martin, G. (2002) 'El proyecto fin de carrera en ingenieria informatica', Prentice Hall, Madrid, ISBN 84-205-3560-5; Spanish translation of Dawson (2000).

- Projectes basats en la investigació.
- Projectes de desenvolupament.
- Projectes d'avaluació.
- Projectes de col·laboració amb la indústria.
- Projectes de resolució de problemes.

www-staff.lboro.ac.uk/~cocwd/index.html

- Projectes de col·laboració amb la indústria.

Realitzats en una empresa, responent a un problema específic d'aquesta. Degut a que és una visió acadèmica no ens podem deixa portar per les necessitats de la indústria i en tot moment em de saber conservar el punt de vista del nostre projecte d'una manera acadèmica.

- Projectes de resolució de problemes.

Pot implicar el desenvolupament de una nova tècnica per resoldre un problema, o per millorar la eficiència dels processos.

Aquest tipus de projectes no s'exclouen entre si, existeixen projectes que engloben aspectes d'alguns d'aquest tipus. Si mirem de buscar una manera assumible per la catalogació dels projectes es per el fet que la naturalesa del projecte es important en el moment de l'elecció del mètode per poder tractar-lo cosa que és el principal objectiu d'aquesta documentació, saber establir la categoria del projecte i aplicar el mètode més adient per la seva estimació.

Si be, es possible trobar nous tipus de projectes també es pot deduir que una combinació d'aquest permet delimitar el tipus de projecte en el que ens trobem.

Aquesta visió acadèmica no és el que en realitat estem buscant, és força general i encara que ens permet catalogar d'una manera fàcil els projectes no ens acosta definitivament a una idea més realista de projecte informàtic, si mes no ens permet una referència inicial sobre la que podem treballar.

2.1.- Característiques diferencials

En general els projectes informàtics presenten un conjunt de característiques que els diferencien d'altres tipus de projectes, com norma general presenten sempre les següents característiques: és concret, excepcional, té una durada limitada i és flexible.

2.1.1.- És concret

Tot projecte informàtic es du a terme per solucionar un problema perfectament identificat, de manera que té un objectiu definit, concret i tangible. Un projecte informàtic no és una activitat genèrica de la que no és coneix el final.

2.1.2.- És excepcional

Els projectes informàtics a banda que pugin ser semblants sempre són únics i diferents d'altres projectes anteriors o futurs. L'analogia amb altre projectes només és un recurs aproximatiu,
degut en que en cada projecte es persegueixen objectius diferents.

2.1.3.- Té una durada limitada

El fet de que una característica sigui la durada limitada es perquè és implícit en un projecte la consecució amb dates finites. També tal com em mencionat anteriorment el recursos han de poder-se dirigir cap altres projectes per tant s'han d'utilitzar durant unes determinades dades, sabent l'inici i el final de la seva utilització.

2.1.4.- És flexible.

El desenvolupament de projectes requereix una mobilitat de recursos assignats. Si be es cert que en la primera qualificació definirem les necessitats, com més llarg sigui un projecte podem tenir la necessitat de modificar-lo durant el seu desenvolupament, el que farà que es modifiqui també el seguiment, control i les planificacions inicials.

Característiques d'un projecte

- És concret.
- És excepcional.
- Té una durada limitada.
- És flexible.

2.2.- Les etapes d'un projecte i el cicle de la vida

Una de les facetes més importants d'un projecte informàtic és que es temporal, el que ens diu que no restarà utilitzant sempre els recursos que se li assignen en un principi.

Els projectes informàtics tenen un desenvolupament viu, es a dir que comencen, fan la seva activitat i acaben. L'objectiu principal de la gestió d'un projecte es controlar les diferents activitats per portar-lo fins a la seva conclusió. Aquesta feina estarà relacionada sempre amb els terminis establerts i amb el pressupost autoritzat.

Tot projecte estarà basat amb un objectiu i per arribar-hi podrem establir diferents etapes seqüencials que ens permetran un seguiment, control del mateix. El tancament es produirà o per la consecució dels objectius o per una interrupció brusca si no s'ha fet una bona qualificació i ens apartem dels marges establerts.

Podem dividir un projecte informàtic en quatre etapes en relació a la seqüència i la seva interrelació:

- Inici del projecte.
- Qualificació del projecte.
- Desenvolupament del projecte.
- Tancament del projecte.

2.2.1.- L'inici del projecte

En aquesta etapa és on s'estableix els requisits i els objectius funcionals generals que s'han d'assolir i, de fet, dona origen al projecte i el fa néixer.

2.2.2.- La qualificació del projecte.

La qualificació d'un projecte és una etapa molt delicada. Durant el procés de creació del document de qualificació s'ha de valorar Quin serà el nostre oferiment a l'hora d'executar la resta del projecte. Una mala configuració d'aquest document pot repercutir directament en els costos relacionats, tant humans com monetaris.

Es necessari una bona estimació del volum de feina així com del recursos necessaris per portar-lo a terme.

2.2.3.- El desenvolupament del projecte.

Durant aquesta etapa del projecte, es molt important el control i el seguiment del mateix. Em de poder valorar en tot moment els possibles desviaments que es puguin produir per poder fer les accions oportunes i poder concloure el projecte en el terminis establerts.

Etapes d'un projecte

- Inici del projecte.
- Qualificació del projecte.
- Desenvolupament del projecte.
- Tancament del projecte.

2.2.4.- El tancament del projecte.

És l'última etapa d'un projecte e indica l'acabament definitiu. Permet fer un balanç de la realització al mateix temps que allibera els recursos que se li havien assignat.

2.2.5.- El cicle de la vida

Anteriorment em definit les etapes per les quals passa un projecte informàtic, aquestes etapes es veuen com un cicle de vida sencer, sobretot si s'hi uneix l'inevitable manteniment de les aplicacions i la retirada al final de la seva vida útil. Mitjançant el cicle de la vida marquem el caràcter evolutiu pales en tots els projectes informàtics.

Existeixen diverses metodologies en molts casos referents a l'enginyeria de programari que parteixen de diferents punts de vista sobre el cicle de la vida d'una aplicació. Aquest punts de vista poden ser utilitzats si tenim un projecte d'administració, encara que cal tenir present que evolutivament han estat dissenyats per el desenvolupament de programari.

cicles de la vida

Hi ha diferents tipus de cicles de vida: en cascada, amb prototip, en espiral, …

El diferents cicle de vida poden ser : en cascada, amb prototip, en espiral etc…

Si examinem en detall el cicle de la vida en cascada veiem que està dividit en varies etapes de manera seqüencial quasi lineal en la construcció per exemple de programari. Totes les etapes han de finalitzar abans de que s'inicii la següent.

Tal com estem intentant des de principi del document busquem utilitzar aquest cicles definits per projectes de construcció de programari en altre tipus com per exemple els d'administració.

Segons un projecte de construcció de programari les etapes per les quals passa serien:

- L'avantprojecte o estudi d'oportunitat
- L'anàlisi del sistema d'informació
- El disseny d'una solució tècnica concreta
- La implementació final del sistema informàtic
  - Desenvolupament
  - Proves
- El manteniment

Mitjançant la corba de cargol podem veure una gràfica del

pas per les diferents etapes del projecte en relació als recursos i el temps

Aquest gràfic ens permet veure com un projecte te un màxim de recursos durant el desenvolupament i les proves i que un tema a tenir present també en l'estimació d'esforç és que el temps de manteniment es molt més que no pas el de construcció (estimació de costos).

2.4.- Els recursos i els costos reals

Com em pogut veure en la definició de les característiques generals i les etapes dels projectes informàtics, els recursos són de vital importància a l'hora de qualificar ,estimar o planificar necessitats. El cap de projecte és el responsable de manega aquest recursos de manera dinàmica per atendre les necessitats dels projectes, no són específics, sinó que el mateix recurs pot estar destinat a diferents projectes en línia segons la seva capacitat. En general podem trobar diferents tipus de recursos:

- El maquinari de les màquines objectiu.
- El maquinari de proves i desenvolupament.
- El programari utilitzat.
- Els recursos humans.

2.4.1.- El maquinari de les màquines objectiu.

Són el conjunt de màquines sobre les que actuarà el nostre projecte una vegada finalitzat, és convenient no perdre mai de vista el lloc final on funcionarà el nostre projecte, una mala estimació del maquinari final pot malmetre el treball de mesos d'un projecte.

2.4.2.- El maquinari de proves i desenvolupament.

És el més semblant al maquinari final i és el que utilitzarem durant el desenvolupament abans de posar-lo en producció.

2.4.3.- El programari utilitzat.

Qualsevol eina utilitzada durant el desenvolupament del projecte ha de ser testat i valorat per la seva utilització. El programari pot ser a nivell de codi , eines orientades a mètode com les eines CASE o qualsevol software creat que pugui ser útil.

2.4.4.- Els recursos humans.

Sempre que decidim engegar un projecte hem d'utilitzar recursos humans, per començar, la qualificació ha de ser feta pel cap del projecte i sempre s'utilitzen tant tècnic com analistes com programadors en els projectes de desenvolupament.

Recursos

- El maquinari de les màquines objectiu.
- El maquinari de proves i desenvolupament.
- El programari utilitzat.
- Els recursos humans.

Una de les necessitats a l'hora de parlar d'aquest tipus de recursos són les aptituds i les actituds dels diferents treballadors utilitzats.

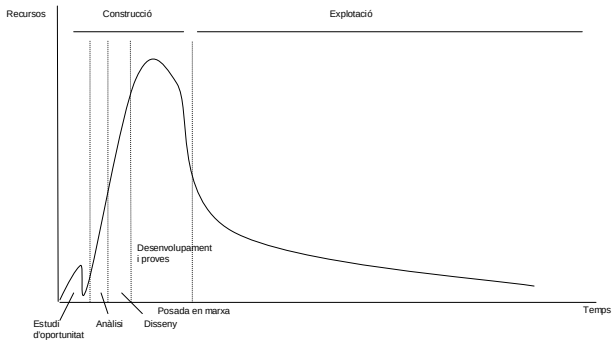
2.4.5.- Els costos reals

Quant optem per plantejar un determinat projecte es evident que en tot moment valorarem els costos reals d'aquest, però de quins costos parlem?.

Hi han molt tipus de costos associats a un projecta que afecten d'una manera considerable la viabilitat, com per exemple el cost de personal, el cost de programari o el cost de maquinari etc…

Una manera més tangible per reflexionar sobre el cost del projecte, és mirar d'estimar els costos per fases de desenvolupament, es cert que en alguna de les fases el cost és alt però el conjunt global és el que determinarà el cost final.

Tenint en consideració el cicle de la vida del projecte ens adonem que el cost real inclouria el manteniment del mateix i és en aquest punt on veiem que qualsevol projecte desenvolupat en disposició del temps que tinguem que mantenir-lo té un cost real molt alt.



Tot i que necessitem molts recursos en la etapa de desenvolupament i proves, la realitat és que el recursos emprats en el manteniment tampoc són negligibles, sempre hi ha necessitat de manteniment ja sigui correctiu, perfectiu o d'adaptació i això durant el llarg període en que està en explotació.

Si miréssim els costos reals d'un projecte tenint especial cura en el manteniment, possiblement no donaríem viabilitat a molts dels projectes que en l'actualitat estan en curs.

El cap de projecte en el moment de fer l'estimació hauria de preveure el cost que esdevindrà de la posada en marxa del projecte i del seu manteniment durant la seva explotació ja que en definitiva són costos que afectaran a la gestió de tot el departament i a futurs projectes.

2.5.- Tipus de projectes una visió empresarial.

Fins ara em pogut veure una manera de catalogar els projectes des de el punt de vista acadèmic, però en realitat el que busquem es poder-ho fer de manera que sigui més fidel a la realitat empresarial de la informàtica. Una manera de buscar aquest tipus la podríem trobar en els possibles projectes que poden ser gestionats per una empresa informàtica o per un departament d'informàtica.

Administració de Servidors	Administració seguretat	Administració d'usuaris
Administració de dades		Administració de xarxa
Comunicacions	Desenvolupament	Administració WEB

Entre d'altres podem trobar dues grans àrees d'actuació en relació a la gestió de projectes:

- Projectes informàtics no relacionats amb el desenvolupament (creació de programari).
- Projectes informàtics de desenvolupament (creació de programari).

2.5.1.- Projectes informàtics no relacionats amb la creació de programari.

Una de les àrees en les cada dia s'han de planificar més nombre de projectes és la d'administració. L'arquitectura de les organitzacions es cada vegada un dels punts on es busquen millores i per això els informàtics tenen una gran oportunitat per crear o millorar la situació actual.

Hi ha una diversitat amplia de projectes relacionats amb l'administració els quals es podem englobar en les següents categories, totes elles sota la responsabilitat del cap d'informàtica :

- Projectes referents a la gestió de servidors.
- Projectes referents a la gestió usuaris.
- Projectes referents a la gestió de xarxes .
- Projectes referents a la gestió dades.
- Projectes referents a la gestió de l'entorn WEB.
- Projectes referents a la gestió de la seguretat

Tipus de projectes visió de mercat

- Projectes informàtics no relacionats amb la creació de programari.
  - Projectes referents a la gestió servidors.
  - Projectes referents a la gestió d'usuaris.
  - Projectes referents a la gestió de xarxes.
  - Projectes referents a la gestió de dades.
  - Projectes referents a la gestió de l'entorn WEB.
  - Projectes referents a l a gestió ó de la seguretat
- Projectes informàtics relacionats amb la creació de programari

2.5.1.1.- Projectes referents a la gestió de servidors.

En totes les organitzacions, ens veien moltes vegades obligats a instal·lar o fer modificacions importants a nivell de servidor, el responsable del projecte té la obligació de valorar la necessitat i estima la viabilitat del projecte. L'estimació del cost de la instal·lació pot fer difícil el manteniment del servei. Em de tenir en compte que

en aquest tipus de projectes poden estar involucrats moltes persones del departament.

2.5.1.2.- Projectes referents a la gestió d'usuaris.

En aquest tipus de projectes podríem englobar tot el que fa referència a l'atenció de les necessitats dels usuaris i el manteniment e instal·lació dels seus equips de sobretaula. Com que el sistema informàtic no deixa de ser una eina per a l'organització, aquesta és una de les parts més visibles i, per tant, cal anar amb molta cura perquè és fàcil que l'usuari tingui problemes amb el funcionament diari del sistema si no preveiem els possibles riscos dels nous projectes en vers de la instal·lació actual.

2.5.1.3.- Projectes referents a la gestió de xarxes.

L'actuació bàsica d'aquest tipus de projectes és la part física de la xarxa: s'ha d'assegurar que l'estimació sigui la correcta tenint cura de l'ampliació progressiva de la xarxa. Degut a l'amplia
oferta en el mercat i a la reducció de costos ens podem beure abocats a reduir les nostres pretensions, reduint les opcions d'ampliació o seguretat del sistema.

2.5.1.4.- Projectes referents a la gestió de les dades.

Quin no ha sentit mai 'la informació és poder', en les organitzacions un dels temes més deixats de la ma, es l'administració de la informació. Moltes vegades la informació només es cerca quant és necessita i en cap cas es té en compte que una bona organització evita duplicitats i dades fictícies. Els
projectes d'administració de dades son els que busquen una millor gestió de la informació i eviten redundàncies de dades.

2.5.1.5.- Projectes referents a la gestió de l'entorn WEB.

Els sistemes WEB (Intranet e Internet) han arrelat en totes les organitzacions de tal manera que s'han convertit en els sistemes comunicacionals més utilitzats per la organització. Una de les raons més importants ha estat la reducció de costos i la facilitat d'implantació.

2.5.1.6.- Projectes referents a la gestió de la seguretat

Quant parlem de seguretat no ho podem fer a la lleugera, la seguretat és un tema molt complex i determina en molts casos l'eficiència de l'activitat informàtica. Els projectes de seguretat són necessaris per la gestió diària, encara que moltes vegades
poden ser els menys valorats degut a que no aportant resultats tangibles.

2.5.2.- Projectes informàtics de creació de programari.

Tant a nivell acadèmic com a nivell real la gestió de projectes de desenvolupament te una importància rellevant. La primera visió
que om te quan es parla de projectes són de desenvolupament, però que hi podem troba dins d'aquest grup?.
Les necessitats informàtiques de una organització han de ser ateses i per això si l'única possibilitat és fabricant programari nou

aquesta és la funció que s'encarrega de fer-ho, per tant la creació dels projectes de desenvolupament de l'empresa està influenciada per la necessitat i els recursos de la mateixa organització. S'ha de sospesar que hi ha altres opcions que els tècnic aconsellen com la compra de programari estàndard o traspassar l'aplicació a una empresa aliena a l'organització, tot i que és el cap d'informàtica que ha de prendre la decisió final mitjançant estudis de viabilitat.

L'aplicació de mètriques o metodologies està molt orientada a l'enginyeria de programari degut a la seva importància en l'evolució de la informàtica.

3.- Abast actual, mètriques i models d'estimació d'esforç.

3.1.- Estimació de costos en la gestió d'un projecte informàtic

La gestió d'un projecte informàtic de gestió té el seu començament amb la qualificació del projecte, que pretén:

- Fer-se una idea del volum de feina una, estimació del cost.
- Planificar les activitats que cal dur a terme, una planificació acurada de les tasques.

La primera d'aquestes etapes es coneix també amb el nom d'**estimació de costos d'un projecte informàtic.** A partir de l'estimació es pot obtenir el pressupost. Una vegada establert el repartiment en el calendari de les diferents tasques s'obté els terminis , aquesta etapa que també es coneix amb el nom de **temps de desenvolupament del projecte**.

Com em vist a la tipificació del projectes aquest poden ser influenciats amb molta mesura per la utilització de recursos degut aquesta utilització els costos que intervenen en un projecte informàtic són variats i en alguns casos complexos.

Segons el tipus de projecte els recursos utilitzats són diferents mentre que en un projecte de servidors els recursos de màquina són molt importants i poden esdevenir un cost gran, en els projectes de construcció de programari el major cost serà el cost de les hores d'anàlisi, disseny, programació i prova que cal utilitzar per a obtenir-lo.

De les dues etapes esmentades en la qualificació del projecte, la de l'estimació de costos i la determinació del volum de feina que s'ha de fer, la segona és clarament la més difícil i complexa.

Existeixen força teories sobre com ajudar-nos a fer aquesta tasca però en molts casos no és suficient per les nostres necessitats o no es del tot aclaridora.

Tom De Marco[1] va deixar clar aquest punt. En un llibre ple de disquisicions interessants i amb molt de sentit comú, De Marco considera que el fet d'esperar obtenir de la bibliografia tècnica dades prou fiables per a fer una bona estimació és una actitud comparable esperar alguna cosa que simplement, no arriba mai.

En paraules de De Marco:

"No hi ha models de cost transportables. Si esperes que algú en un altre lloc desenvolupi un conjunt de fórmules que puguis fer servir per a preveure el cost en la teva pròpia instal·lació, probablement hauràs d'esperar per sempre."

T. De Marco (1982, pàg. 155).

[1] "El dilema de l'estimació" de l'obra *Controlling Software Projects* (cap. 2, pàg. 9-17). T. De Marco (1982). Englewood Cliffs: Yourdon Press (Prentice-Hall).

Aquesta apreciació que podem donar com certa ens abocaria directament a pensar que l'única opció que tenim és la nostre experiència, per tant és imprescindible disposar de dades concretes sobre projectes anteriors fets en una instal·lació determinada. Però clar no sempre tenim projectes anteriors o més simple algun dia tenim que començar a gestionar projectes i estimar-ne els costos be, no ens desesperem, si no es tenen aquestes dades, s'han de fer servir les de la bibliografia tècnica[2], encara que no és sempre raonable fiar-se'n gaire. Potser per això, el mateix De Marco proposa dues definicions successives del que és una estimació que són molt realistes:

1) **Definició implícita d'estimació**: una estimació és la predicció més optimista que té una probabilitat no nul·la d'arribar a ser certa.

2) **Definició proposada per De Marco**: una estimació és una predicció que té la mateixa probabilitat d'estar per sobre que d'estar per sota del resultat real.

Visió pessimista…

En resum, estimar la càrrega de feina que cal portar a terme per a la construcció de programari d'aplicació és una tasca que normalment ha de fracassar.

Encara que les expectatives no són gaire bones, per començar, cal fer una estimació i, deixant ben clar que en aquests aspectes val més l'experiència que les fórmules però em de seguir esbrinant per trobar models o mètriques que ens puguin ajudar en la nostre tasca de gestió.

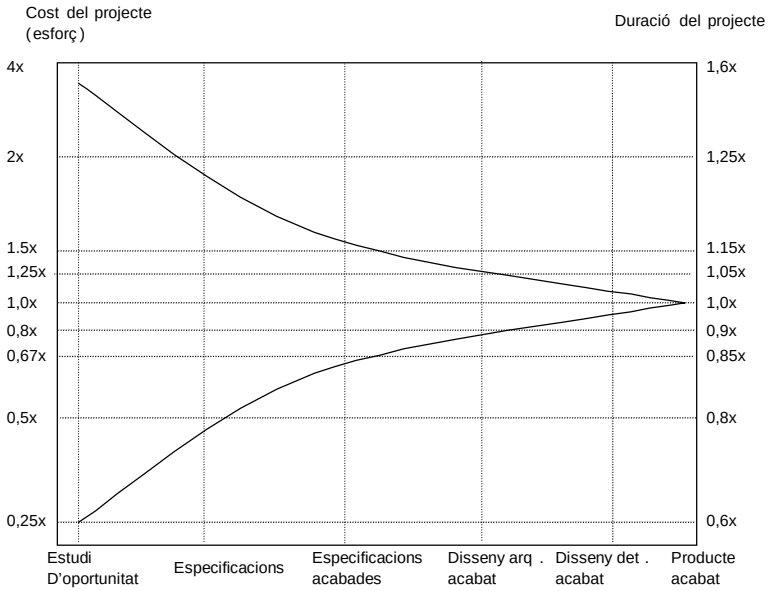
3.1.1.- Moment de l'estimació i grau d'exactitud

Es de sentit comú veure que quan hem de fer la primera estimació no disposem de totes les especificacions del projecte que em de dur a terme.

En un principi no disposem de totes les funcionalitats que caracteritzen el projecte, és en etapes posteriors on aquestes especificacions quedaran determinades i podrem conèixer l'abast real del projecte que gestionem.

Mirem de calcular de manera que ens orienti quina és la desviació que es presenta entre l'estimació del volum de feina que cal fer i el volum que s'ha de fer realment al final. El gràfic de la pàgina següent mostra com, a mesura que avança el projecte informàtic i se'n coneixen amb més detall les especificacions, el marge d'error va disminuint:

[2] Per exemple, les productivitats de l'ordre de 10 a 16 LOC per persona - dia o 2 hores per a implementar un punt de funció.



Font: Gràfic adaptat de "Cost Models for Future Life Cycle Processes: COCOMO 2.0" de B.W. Boehm i altres (1995).

A la gràfica podem veure com els errors d'estimació inicials poden se de l'ordre de quatre vegades per sobre o per sota de la que al final serà la carrega real de feina (escala d'ordenades esquerra).

La duració estimada te un marge d'error que va entre 160% i el 60% de la duració real (escala d'ordenades dreta).

3.2.-Mètriques per la gestió d'un projecte informàtic

Es evident que per poder fer una estimació dels costos d'un projecte informàtic tant dels que estan lligats a la construcció de programari com els que no hem de poder diferenciar unitats mesurables que puguin relacionar la producció amb l'esforç del treball o el cost monetari . És aquí on les mètriques ens poden ajudar a resoldre aquesta mesura.

En un principi parlarem de la mesura de projectes de programari i més endavant intentarem esbrinar o tornar a definir aquestes mètriques per altre tipus de projectes informàtics.

Es pot dir que tota magnitud física es pot mesurar, però per exemple el programari és, sobretot, el resultat d'una activitat bàsicament intel·lectual. La dificultat de mesurar el programari ha fet néixer diverses mètriques o unitats de mesura que intenten quantificar diferents aspectes que poden ser importants en el programari.

Hi ha tres conceptes que convé especificar de manera diferenciada:

Definició de mètrica

És una assignació de valor a un atribut d'una entitat pròpia del programari.

l'*Standard Glossary of Software Engineering Terms* de l'IEEE.

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

- Quan parlem d'un **atribut**, fem referència a una determinada característica que pretenem mesurar i quantificar[3].

- Quan parlem d'un **producte** ens referim, per exemple, a un codi, un disseny, una especificació, etc.

- Quan es parla d'un **procés** es fa referència a una etapa de la construcció del programari i a la manera com s'ha portat a terme: un disseny, unes proves, etc.

En general les diverses mètriques intenten avaluar tres grans magnituds: la **qualitat**, la **grandària** i la **productivitat.**

Quant volem fer l'estimació de costos, les mètriques de productivitat són les que tenen més rellevància en la qualificació d'un projecte informàtic. L'elecció d'una mètrica en concret acostuma a ser un aspecte cabdal en els diversos models d'estimació.

3.2.1.-Mètriques per la construcció de programari

3.2.1.1.- Mètriques del producte i del procés

Inicialment podem subdividir totes les moltes mètriques que hi ha en dos tipus següents:

1) Les **mètriques del producte**, que mesuren diversos aspectes del programari obtingut, sovint a partir de codi font expressat en un llenguatge informàtic determinat[4].

2) Les **mètriques del procés**, que intenten determinats atributs que fan referència a l'entorn de desenvolupament del programari i tenen en compte la manera com es construeix.

Les mètriques es poden fer servir soles o combinades, és a dir, com a senyals que permeten una visió resumida del producte de programari o del procés de la seva construcció.

1) Els **indicadors de producte**, referits, evidentment, al programari obtingut, poden servir per a avaluar la qualitat del procés de construcció de programari, sempre de manera comparativa amb estàndards de mercat o amb nivells assolits en altres productes.

2) Els **indicadors de procés** han de permetre conèixer l'eficàcia d'una instal·lació simplement per comparació amb els indicadors que és capaç d'assolir respecte dels que la literatura tècnica publica.

3.2.1.2.- Mètriques de qualitat i de productivitat

Podem resumir Les unitats de mesura per a quantificar el programari en qualitat del producte i productivitat del procés.

[3] Com podrien ser l'esforç, la fiabilitat, el manteniment, la qualitat o la grandària.
[4] Llenguatges de programació de disseny, d'especificació, etc…

Les mètriques de qualitat s'associen a uns determinats atributs mesurables, no sempre evidents per a reconèixer la presència o l'absència de qualitat.

De vegades, les mètriques de qualitat formen part del que s'anomenen *mesures indirectes del producte*, juntament amb les mètriques de funcionalitat, complexitat, eficiència, fiabilitat de manteniment i tantes altres capacitats en certa manera difícils de mesurar objectivament.

Les mètriques de productivitat recullen l'eficiència del procés de producció de programari i relacionen el programari que s'ha construït amb l'esforç que ha costat fer-lo.

En la terminologia de l'estàndard de mètriques de productivitat de programari de l'Institut d'enginyers elèctrics i electrònics (*IEEE Standard for Software Productivity Metrics*) de 1993, es parla de **primitiva** per a fer referència al nivell més baix on es recullen les dades. Evidentment, hi ha primitives de dos tipus:

- **De sortida**[5], que recullen les característiques finals del programari.
- **D'entrada**[6], que mesuren el que ha calgut tenir i fer servir per a construir el programari.

Les unitats de mesura com a indicadors

Sovint les diverses unitats de mesura es combinen en indicadors resumits, dels quals serien exemples, entre d'altres, els següents:

- Línies de codi per persona i dia (indicador de productivitat).
- Hores per a 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 documentació).

3.2.1.3.- L'esforç i la mesura de la productivitat

Per tractar la productivitat en el procés de construcció de programari d'aplicació, amb independència de quines siguin les primitives de sortida utilitzades (generalment, les línies de codi o els punts de funció que estudiarem més endavant), les primitives d'entrada intenten sovint mesurar l'esforç o el cost que representa el fet de construir-lo. La productivitat s'expressa finalment en una ràtio que relaciona les primitives d'entrada i de sortida.

Definició de qualitat d'un producte segons ISO 8402

Conjunt de funcionalitats i característiques d'un producte o servei que se centren en la seva capacitat de satisfer les necessitats, ja sigui implícites o bé explicitades clarament d'un client o usuari.

La mesura habitual del cost és, la quantitat de diners que costa produir un programari determinat i les mesures d'esforç sempre són les feines que fa un professional (qualificació mitjana, bona formació i capacitat de treball mitjana) en una determinada unitat de temps (home – mes, home – any….)

3.2.1.3.1.- Els problemes terminològics de l'home - mes

En els projectes informàtics podem trobar diverses denominacions de la unitat d'esforç, com ara les següents:

1) En primer lloc, es va parlar d'home-mes com la tasca que feia un home durant un mes de feina (MM *man-month).*

2) Més endavant, va semblar que la denominació era clarament sexista i s'hi ha buscat altres noms com aquests:

[5] En anglès, *output primitives*.
[6] En anglès, *input primitives*.

- **Persona - mes:** un mes de feina d'una persona de l'equip amb independència del gènere.
- ***Staff-month:*** un mes de feina d'una persona de l'equip de projecte[7].
- ***Engineering-month:*** un mes de feina en l'activitat d'enginyeria del programari, que inclou, com ja sabem, l'anàlisi, el disseny, la programació i les proves per a produir una determinada aplicació o un sistema d'informació.

3.2.1.4.- Mètriques orientades a la grandària i a la funció

La **grandària del programari**, construïda a partir de les línies de codi que aquest inclou, és una mesura tradicional d'avaluar el volum de feina.
En un principi és possible creure que les línies de codi podrien ser una mesura valida per valorar l'esforç però el cert és que no està relacionada l'efectivitat ni el temps de consecució d'un codi amb la quantitat de línies.

Degut a que la quantitat de línies no poden donar-nos una mètrica consistent s'han buscat noves que són més fiables no pas veien la grandària sinó la dificultat per aconseguir-los. Un exemple prou conegut és el dels punts de funció d'Albrecht,

3.2.1.4.1.- Les línies de codi

Durant molt de temps la mesura de grandària que més 'ha utilitzat és la de les línies de codi del programari final obtingut. Hi ha diverses denominacions als text especialitzada com per exemple les següents:

a) **LOC** (Lines of Code): línies de codi. És la més habitual i antiga.

b) **KLOC**: milers de línies de codi. És la unitat de mesura que adopten la majoria de models clàssics d'estimació de costos en la qualificació d'un projecte informàtic.

c) **DSI** (Delivered Source Instructions): instruccions de codi font realment lliurades, i el seu múltiple **KDSI**, que van sorgir perquè es tingués en compte que, en els nous entorns de desenvolupament i construcció de programari, bona par de les línies de codi no sempre les ha escrit l'equip del projecte, sinó que les han generat les eines de productivitat de l'entorn de programació.

d) **NCSS** (Non Comment Source Statements): línies de codi font sense tenir en compte els comentaris, i el seu múltiple (**KNCSS**), pel fet de considerar que en un bon procés de construcció els programes inclouen línies de comentari o que una línia de tractament es pot escriure en diverses línies de codi, per a augmentar la llegibilitat i millorar el manteniment del programari.

e) **NSLOC** (New Source Lines of Code): noves línies de codi font, tal com es fa el model COCOMO 2 perquè es tingui en

[7] En anglès, *staff*.

compte que només cal comptar les línies de codi noves sense comptar les que incorpori automàticament l'entorn de programació, o, el més important en aquest cas, el fet que part del codi final pot procedir de la reutilització de rutines o objectes ja disponibles que no s'han de tornar a escriure, però que s'incorporen al projecte informàtic en curs.

El fet que les línies de codi no siguin un valor mesurable fiable és que els llenguatges de programació han anat evolucionant i cada vegada fan falta menys línies per aconseguir el mateix.
En el seu llibre de 1986, l'especialista Caper T. Jones oferia dades que il·lustren diversos aspectes de la productivitat associada als diferents llenguatges de programació:

forç per a desenvolupar la mateixa quantitat de línies de codi en diversos llenguatges			
atges	Grandària (LOC)	Esforç (persona-mes)	LOC per persona-any
blador	10.000	40	3.000
ssemblador	10.000	42	2.857
	10.000	44	2.727
.	10.000	44	2.500
	10.000	51	2.354
	10.000	52	2.308
	10.000	53	2.264

Per tant, possiblement és més adient la taula següent, també extreta del llibre de Jones, que ofereix resultats semblants, però que parteix de l'esforç que cal per a desenvolupar, en diversos llenguatges, uns programes que implementin una mateixa funcionalitat.

orç i línies de codi per a desenvolupar la mateixa funcionalitat en diversos llenguatges			
uatges	Grandària (LOC)	Esforç (persona-mes)	LOC per persona-any
blador	10.000	40	3.000
assemblador	6.666	28	2.856
	5.000	22	2.727
L	3.333	26	2.500
l	2.500	13	2.307
	2.222	12	2.222
	2.000	11	2.181

Aquestes dades vénen dels anys setanta a vuitanta, i no recullen el que es pot aconseguir amb llenguatges de programació de productivitat més elevada o amb les facilitats dels nous entorns de programació i les seves ajudes.

L'evolució general a permès que hi hagin llenguatges molt fàcils d'utilitzar que tenen les avantatges de l'orientació a objectes i la programació virtual. Nascuts en el món de la microinformàtica per a ajudar a la realització de programes amb interfície visual sota el paradigma WIMP (*windows, icons, mouse* i *pop-up menú*; és a dir, finestres, icones, ratolí i menús desplegables), constitueixen la manera més moderna i més evolucionada del que s'ha

anomenat RAD (*rapid application development*, és a dir, el desenvolupament ràpid d'aplicacions).

Tots aquest nous llenguatges encara no poden ser degudament avaluats degut a que encara no hi ha dades fiables que ens permetin determinar el seu esforç.

Línies de codi i el conjunt del projecte informàtic

Les LOC de les que parlem són les línies exactes que conformen la creació del programari però a més a més incorporen la mesura de productivitat avaluant les eines que han fet falta per arribar a la codificació.

> Aquestes dades són mostrades per Caper T. Jones en un dels seus llibres (1986) les que ens poden ajudar a aclarir el significat de les LOC. Jones parla de la *productivitat aparent d'una persona* a partir de les LOC que es poden implementar per persona i any. Però cal saber quina és la feina que incorpora cada una de les mesures o estàndards.

3.2.1.4.2.- Els punts de funció

Hi ha diversos models d'estimació de les càrregues d'un projecte informàtic. La majoria fan servir com a mesura de les primitives de sortida la mesura dels punts de funció que va definir Albrecht en un primer article que data de l'any 1979 i que va ser revisat (i relacionat també amb la mètrica de les línies de codi) l'any 1983 pel mateix Albrecht, ajudat per Gaffney.

Des de l'any 1984, l'anomenat *grup internacional d'usuaris dels punts de funció*, IFPUG (de la denominació anglesa *International Function Points User Group*) publica periòdicament la manera correcta d'avaluar els diversos aspectes discrecionals del recompte de punts de funció d'Albrecht. La versió 3.0, que data de l'any 1990, i la 4.0, que és de 1994, són les més utilitzades avui i la diversa literatura tècnica actual sobre punts de funció hi fa referència sovint[8].
Avui dia la majoria d'especialistes estarien d'acord a afirmar que la mètrica dels punts de funció és la més utilitzada. El treball de l'IFPUG[9] la posa en certa manera al dia, mentre que les mètriques basades en les línies de codi no sempre s'han adaptat bé als nous llenguatges de programació o a les eines RAD.

L'IFPUG ha decidit mantenir sense alterar l'estructura de la mètrica i això no en facilita la utilització avui malgrat l'evolució de l'entorn informàtic. En concret, l'IFPUG recomana, seguint les directriu que estableix, construir procediments normalitzats i uniformes a

[8] A.J. Albrecht; J.E. Gaffney Jr. (1983). "Software Function, Source Lines of Code, and Development Effort Prediction: A. Software Science Validation". IEEE Transactions on Software Engineering (vol. SE-9, núm. 6, novembre, pàg. 639-648).
[9] A l'adreça www.ifpug.org es pot trobar informació de la versió més actual de l'IFPUG a Internet, tot i que per a obtenir determinades dades o manuals cal ser-ne soci.

cada instal·lació per a procedir al recompte dels punts de funció sempre d'una mateixa manera homogènia.

Caldrà establir per a cada instal·lació quina és la ràtio de productivitat (punts de funció per persona - mes o per hora de treball) que són imprescindibles per a una estimació i qualificació correctes de projectes informàtics futurs.

Determinació dels punts de funció

El recompte dels punts de funció s'elabora a partir de determinades característiques funcionals, que poden ser de dades o de transacció:

1) Les **característiques funcionals de dades** són les que presentem a continuació:

a) **Fitxers lògics interns (LIF)**: grup de dades inter-relacionades lògicament i que poden ser identificades clarament fins i tot pels però que, en qualsevol cas, sempre es mantenen dins dels límits de l'aplicació, és a dir són internes i pròpies de l'aplicació. Són els fitxers de l'aplicació (o, si es vol, les taules de la base de dades).

b) **Fitxers d'interfícies externs (EIF)**: grup de dades inter-relacionades lògicament i que poden ser identificades pels usuaris, o a vegades informació de control però, en aquest cas, procedeixen de fora dels límits de l'aplicació i es mantenen al marge de l'aplicació. Representen els fitxers o les taules que l'aplicació utilitza però que no crea ni manté.

2) Les **característiques funcionals de transacció** són les següents:

a) **Entrades externes (EI):** fan referència als tractaments que processen dades o informació de control que s'introdueix a l'aplicació des de fora dels seus límits. Són, evidentment, les entrades a l'aplicació.

b) **Sortides externes (EO):** qualsevol procés elemental que generi dades o informació de control que surti fora dels límits de l'aplicació. Equival a les sortides que genera l'aplicació.

c) **Consultes externes (EQ):** procés elemental format per una combinació d'entrada i sortida que permet la recuperació de dades. La sortida no ha de contenir dades derivades i es poden utilitzar els fitxers lògics interns. Són les consultes internes de l'aplicació, que sovint es resolen amb consultes als fitxer o taules propis.

Seguint una sèrie de directrius més o menys objectives que avui elabora i manté l'IFPUG, cadascuna d'aquestes característiques funcionals queda definida com si fos de complexitat simple, mitjana o complexa. En la taula següent es mostra la ponderació de cada característica funcional i la disposició dels càlculs per a obtenir, gràcies a pesos de ponderació fixats per Albrecht al final

dels anys setanta, el que s'anomena **total de punts de funció sense ajustar (TUFP)**

Punts de funció sense ajustar (TUFP)					
Tipus	**Descripció**	**Complexitat**			**Total**
		Simple	**Mitjana**	**Complexa**	
EI	Entrades externes	— x 3	— x 4	— x 6	—
EO	Sortides externes	— x 4	— x 5	— x 7	—
LIF	Fitxers lògics interns	— x 7	— x 10	— x 15	—
EIF	Interfícies lògiques externes	— x 5	— x 7	— x 10	—
EQ	Consultes externes	— x 3	— x 4	—x 6	—
Total de punts de funció sense ajustar (TUFP)					—

El mateix Albrecht ja va pensar que les característiques funcionals indicades no esgotaven totes les possibilitats de definir funcionalment un programari d'aplicació. Per això va afegir que calia considerar també les característiques generals del sistema. En la formulació establerta l'any 1979 (i mantinguda encara per l'IFPUG sense modificacions) hi ha catorze característiques funcionals i s'avalua cadascuna de 0 a 5 (el valor mitjà és, concs, 2, 5).

La taula següent detalla les característiques generals del sistema i una disposició adient al seu càlcul:

Característiques generals del sistema: Grau total d'influència (TDI)		
ID	Característiques del sistema	Total
C1	Comunicació de dades	—
C2	Funcions distribuïdes	—
C3	Rendiment	—
C4	Configuració molt utilitzada	—
C5	Freqüència de transaccions	—
C6	Entrada on-*line* de dades	—
C7	Eficiència d'usuari final	—
C8	Actualització on-*line*	—
C9	Processos complexos	—
C10	Reusabilitat	—
C11	Facilitat d'instal·lació	—
C12	Facilitat d'operació	—
C13	Instal·lació múltiple	—
C14	Facilitat de canvis	—
Grau total d'influència (TDI)		

Aquestes característiques generals afecten el resultat dels punts de funció de manera que la suma dels valors d'aquestes(que està sempre entre 0 i 70) s'anomena **grau total d'influència (TDI).**

Amb el valor del grau total d'influència es determina l'**ajust per la complexitat del procés (PCA)** segons la fórmula següent:

$$PCA = 0{,}65 + (0{,}01 \cdot TDI)$$

Finalment, els **punts de funció (FP)** s'obtenen ajustats i definitius segons el càlcul següent:

FP = TUFP · PCA

on PCA és el valor de l'expressió anterior.

Malgrat que l'IFPUG ha decidit mantenir les mateixes característiques i ponderacions que va triar Albrecht sense modificar-ne els pesos de ponderació periòdicament va publicant guies per a ajudar a la determinació dels valors de les característiques generals i, també, del grau de complexitats de les característiques funcionals.

Com exemple algunes recomanacions importants de l'IFPUG, extretes de la versió 4.0 (1994) del manual de l'IFPUG:

1) La **complexitat funcional sobre els fitxer** (tant els LIF com els EIF) es determina a partir de dos conceptes:

- El nombre d'elements, o camps de dades, d'un fitxer (DET[10]).
- El nombre de registres elementals diferents (RET[11]).

La complexitat de les característiques funcionals surt, en aquest cas, d'una taula com la que presentem a continuació:

Mesura de la complexitat dels LIF i els EIF			
	1 a 19 DET	20 a 50 DET	51 DET o més
1 RET	Baixa	Baixa	Mitjana
2 a 5 RET	Baixa	Mitjana	Alta
6 RET o més	Mitjana	Alta	Alta

2) De manera similar, la **complexitat de les característiques generals** es pot avaluar a partir de les guies que publica l'IFPUG. Per exemple, a la versió 4.0, la primera característica funcional (comunicació de dades) es determina amb els valors següents:

Característica funcional: comunicació de dades	
0	L'aplicació és purament *batch* o s'executa en un únic PC
1	L'aplicació és *batch,* però té entrada de dades o impressió de remotes
2	L'aplicació és *batch,* però té entrada de dades i impressió de remotes
3	L'aplicació inclou entrada de dades *on-line* o TP (teleprocés) *front-end* per a un procés *batch* o un sistema de consulta
4	L'aplicació és més que un *front-end*, però suporta només un tipus de protocol de comunicacions de TP
5	L'aplicació és més que un *front-end*, i suporta més d'un tipus de protocol de comunicacions de TP

[10] *Data Element Type*
[11] *Record Element Type*

Els punts de funció i la productivitat

Per a tenir dades de la productivitat a partir dels punts de funció, s'acostuma a utilitzar una ràtio de productivitat que indica el nombre d'hores de feina (sempre d'un professional mitjà) que calen per a implementar un punt de funció.

Una productivitat que tingui entre dues o tres hores per punt de funció podria ser la més habitual en la majoria de les instal·lacions modernes.

3.2.1.4.3.- Relació entre punts de funció i línies de codi

Es evident que punts de funció i les línies de codi són diferents, però és possible trobar una mena d'equivalència entre els uns i les altres. De fet, ja hi ha milers de projectes informàtics que s'han mesurat fent servir punts de funció i, també, línies de codi, fet que ha permès d'obtenir ràtios de conversió d'una unitat a l'altra.

S'han publicat diverses equivalències entre FP i LOC i la que proposava Caper T. Jones és la següent:

LOC per punt de funció	
Llenguatge	**LOC/FP**
Asemblador	320
Macroassemblador	213
C	150
COBOL	106
Fortran	106
Pascal	91
RPG	80
PL/I	80
Ada	71
BASIC	64

Font C.T. Jones, 1986

Bernard Londeix[12] intenta buscar formules d'equivalència, que en el cas concret del llenguatge COBOL proposa la fórmula de conversió següent:

$$LOC = 118{,}7 \cdot FP - 6{,}490$$

Però, malgrat l'interès per l'exactitud, val a dir que les imprecisions en la determinació de les LOC[13] i, sobretot, dels FP (tot i els estàndards i les recomanacions de l'IFPUG), fan molt raonable la proposta del mateix Caper T. Jones.

> La proposta de Caper T. Jones en la seva columna mensual a la revista IEEE *Computer.* L'any 1996, concretament el mes de març, establia l'equivalència següent:
>
> "Regla 1: un punt de funció = 100 sentències de codi font lògic."

[12] B. Londeix (1987). *Cost Estimation for Software Development.* Reading (Massachusetts): Addison-Wesley.

[13] Conversió de LOC a FP: 1) Per a llenguatges de procediment, com C, COBOL, Fortran o Pascal:100 LOC per FP. 2) Per a llenguatges orientats a objecte i generadors d'aplicacions: 20 LOC per FP.

En aquell mateix article, aquest expert justificava l'equivalència d'aquesta manera:

"La ràtio entre sentències de codi font lògic sense comentaris al punt de funció va des de més de 300 sentències per punt de funció per als llenguatges bàsics d'assemblador fins a menys de 20 per als llenguatges orientats a objectes i la majoria dels generadors de programes. Com que els llenguatges de procediments (*procedural)* com C, COBOL, Fortran i Pascal són propers a una relació de 100 a 1, aquest valor pot servir com a factor de conversió groller."

"Aquesta regla té un gran marge d'error i hi calen correccions significatives quan es tracta de llenguatges de programació orientats a objectes, generadors d'aplicacions o aplicacions on es faci servir una quantitat significativa de codi reutilitzat."

C.T. Jones, (1996, pàg. 116).

3.2.2.- Mètriques aplicables a la gestió de projectes informàtics no de construcció de programari.

Fins ara hem valorat les possibles mètriques referents a gestió de projectes de creació de programari. Hem pogut veure que en la construcció de programari les mètriques podien ser dividides en mètriques de producte i mètriques de procés. Quant no parlem de projectes de construcció de programari, podríem també dividir les mètriques en aquest dos grans grups degut que en aquest projectes també volem mesura la qualitat la grandària i la productivitat i utilitzant els mateixos recursos a la vegada que és poden utilitzar les etapes i cicles de la vida esmentats, però no conceptualment com a la creació de programari.

En el nostre cas la qualitat i la productivitat tenen el mateix sentit però la grandària és la quantitat d'usuaris afectats per el projecte i les hores utilitzades.

En la actualitat la valoració de l'esforç és un camp força obert que no ha estat perfectament definit. La gestió dels projectes que no són de creació de programari queden força oberts a l'experiència però no per aquest motiu em de deixa de banda l'experiència amb els projectes de programació en la tipificació de mètriques. De les tres mètriques definides podem utilitzar en el nostre cas és la següent:

Mètriques de qualitat i de productivitat

Podem resumir Les unitats de mesura per a quantificar el projecte en qualitat del producte i productivitat del procés.

Les mètriques de qualitat s'associen a uns determinats atributs mesurables, no sempre evidents per a reconèixer la presència o l'absència de qualitat.

Les mètriques de productivitat recullen l'eficiència del procés realitzat i relacionen el sistema que s'ha construït amb l'esforç que ha costat fer-lo.

L'esforç i la mesura de la productivitat

- Persona - mes: un mes de feina d'una persona de l'equip amb independència del gènere.
- Staff-month: un mes de feina d'una persona de l'equip de projecte .

- Engineering-month: un mes de feina en l'activitat d'enginyeria del programari, que inclou, com ja sabem, l'anàlisi, el disseny, la implantació i les proves per assolir un determinat projecte.

Quant un canvi influeix en el treball genera un cost de parada en cada un dels usuaris anomenat PDI (preu d'incompliment) de servei, la tasca és minimitzar el màxim aquest preu i reduir al màxim el cost del projecte.

3.3.- Models per la gestió d'un projecte informàtic.

3.3.1.- Els mètodes d'estimació possibles segons Barry W. Boehm

En el llibre de Barry W. Boehm (1981), que ha esdevingut un clàssic sobre el tema, ensenya les set maneres diferents de fer una estimació inicial de costos d'un projecte informàtic. Alguns d'aquest mètodes podem semblar poc raonables però la realitat del mercat els hi dona sentit, si més no tots no són del tot recomanables.

A continuació s'exposen els set mètodes que esmenta Boehm:

1) **Utilització de models:** Boehm els anomena ***models algorísmics*** i proporcionen un o més algorismes que donen una estimació del cost del programari com a funció d'un nombre de variables determinat que es considera que influeixen en aquest cost. Aquests models, com veurem, poden ser també de base estadística, teòrica o compostos.

2) ***Judici d'experts en el tipus de projecte*:** mètode que suposa basar-se fonamentalment en l'experiència d'un o més professionals experts que ja han actuat diverses vegades com a caps en projectes prou semblants al que ens preocupa.

3) ***Analogia amb un altre projecte semblant*:** estratègia que pretén emparar-se en les dades disponibles obtingudes en un altre projecte de construcció de programari que sigui en certa manera anàleg al que ens ocupa en aspectes clau, com la tecnologia, el tipus d'aplicació, el personal que hi intervé, etc.

4) ***Utilitzar tots els recursos disponibles***: cas habitual que es dóna quan no es gestiona un projecte informàtic i s'hi destinen els recursos que semblen necessaris a cada moment del desenvolupament. Respecte a aquest punt, Boehm fa una referència explícita al que es coneix com la *llei de Parkinson,* segons la qual el treball sempre s'expandeix fins a ocupar tots els recursos disponibles. Aquesta llei és fruit de l'observació i l'enginy de Cyril Northcote Parkinson.

5) ***Preu guanyador***: estratègia que proposa simplement una estimació de càrregues (esforç, cost i/o temps de desenvolupament) que faci que l'oferta sigui irresistible, amb independència de si després es pot acomplir o no. O s'iguala automàticament el cost al valor més baix per a aconseguir un contracte, o es tria automàticament un temps de desenvolupament i uns terminis que, amb independència de la

Barry W. Boehm

TRW Professor of Software Engineering, Computer Science Department
Director, USC Center for Software Engineering.

1. Characteristics of Software Quality, North Holland, with J.R. Brown, H. Kaspar, M. Lipow, G. McLeod, and M. Merritt, 1978.
2. Software Engineering Economics, Prentice Hall, 1981.
3. Software Risk Management, IEEE Computer Society Press, 1989.
4. "Anchoring the Software Process", IEEE Software, July 1996.

http://sunset.usc.edu/Research_Group/barry.html

Una estratègia molt utilitzada

Malauradament l'estratègia de preu guanyador és una de les més utilitzades, la competència del mercat fa que les ofertes siguin baixes assumint els possibles problemes que hi puguin haver en la confecció del projecte.

feina que cal fer, permetin al producte ser el primer a arribar al mercat.

6) ***Estimació global descendent***[14]**:** mètode que intenta obtenir una dada única i global per a tot el projecte informàtic a partir de diverses propietats globals del producte final (grandària, complexitat, dificultat tècnica, nivell de qualitat i certesa, etc.) que, després es va descomponent.

7) ***Descomposició en activitats (WBS) i estimació ascendent***[15]**:** mètode que descompon el conjunt del projecte en diverses activitats[16] i, una vegada estimat l'esforç per a cadascuna d'aquestes, obté per agregació l'esforç total del projecte.

Els sis primers esmentats per Boehm representen mètodes estratègics i tàctics que en moltes ocasions poden donar una estimació única de total de projecte. Com podrem veure més endavant, el mètode més operatiu és el setè, que permet, a més, obtenir un desglossament del projecte informàtic en activitats que ens serà molt útil tant en la planificació com en el seguiment i control del projecte.

La majoria dels mètodes de què parlen els llibres tracten l'estimació de costos per mètodes i models algorísmics o amb base estadística.

3.3.2.- Models d'estimació en la construcció de programari.

Els models d'estimació tenen com a base la de proporcionar sistemes i mètodes generals per poder fer l'estimació de costos en la construcció de programari, en el apartat 4.2.3 Aplicació dels models d'estimació en la gestió d'altres tipus de projectes informàtics, mirarem d'aplicar aquesta filosofia en la valoració de projectes que no siguin de creació de programari.

Quant encara no es disposava de models, s'utilitzava l'experiència del cap de projecte, que sovint basava les seves estimacions en una experiència pròpia no sempre objectiva o reflectida en dades concretes.
La introducció dels programes de mètriques de programari permet disposar de dades reals de projectes ja fets, d'aquesta manera es pot obviar la subjectivitat personal i intentar expressar en models i fórmules tot allò que l'estudi estadístic o teòric de les dades disponibles ens permet deduir.

Els diversos **models d'estimació de costos i/o esforços en la construcció de programari**[17] es poden dividir en cinc grans grups principals: models amb base històrica, amb base estadística, teòrics, compostos i basats en estàndards. A continuació els presentem breument:

[14] En anglès, *top-down*.
[15] En anglès, *bottom-up*
[16] En anglès *Work Breakdown Structure*.
[17] En la denominació anglesa, *Costo Estimating Models.*

1) Els **models de base històrica** són els més antics i, en certa manera, primitius. Sovint es basen en l'analogia amb altres projectes semblants i es fonamenten quasi exclusivament en l'experiència professional (la "història") dels qui efectuen l'estimació.

2) Els **models de base estadística** intenten superar l'experiència històrica concreta d'uns quants professionals i, a partir de l'estudi estadístic de les dades reals disponibles mesurades en un conjunt més o menys nombrós de projectes ja acabats, obtenen fórmules que relacionen les diverses unitats de mesura del programari, sovint les línies de codi (LOC) i l'esforç (generalment mesurat en home- mes).

3) Els **models de base teòrica** proporcionen un altre enfocament. Aquests models, més que basar-se en dades estadístiques disponibles no sempre prou abundants i/o generalitzables, el que fan és parir d'una sèrie d'idees generals sobre el procés de construcció de programari i, sobre aquesta teoria, elaboren fórmules que relacionen diverses mètriques de programari.

4) Els **models compostos** han arrelat a partir dels treballs de Barry W. Boehm. Aquests models intenten obtenir els avantatges dels dos sistemes anteriors: estadístics i teòrics.

   És a dir, es parteix d'una sèrie de plantejaments teòrics, i es complementen o corregeixen amb dades estadístiques obtingudes de projectes reals ja acabats.

5) Com que massa sovint els models esmentats abans són poc fiables i ofereixen un grau d'error no gens menyspreable, la pràctica professional ha fet que moltes instal·lacions[18] hagin adoptat un sistema més simple i menys complicat: l'**elaboració d'estàndards de productivitat** obtinguts en projectes anteriors de la mateixa instal·lació. Per a fer l'estimació de projectes nous es parteix d'aquests estàndard. També al marge dels estàndards propis de cada instal·lació, a la bibliografia tècnica es troben les que podríem anomenar *recomanacions practiques a cop d'ull*[19]*,* que han estat elaborades per especialistes prestigiosos i que poden actuar com a referència general futura per als estàndards propis d'una instal·lació que comença.

Gran part dels models disponibles s'han elaborat durant els anys setanta i vuitanta, això no els fa propers a l'actualitat on els nous sistemes de programació proporcionen molts més recursos (llenguatges visuals i orientats a objectes, noves eines d'ajuda, etc.) Aquest models d'estimació més algorísmics poder actuar sempre com a punt inicial de l'estimació que es vol fer.

Les línies de codi (LOC) són utilitzades com a base per obtenir altres mesures com per exemple l'esforç mesurat en persona – mes, els mesos de construcció el nombre de experts necessaris, el nombre d'errors esperats etc…

[18] Generalment, les instal·lacions més grans i amb més experiència.
[19] *Rules of thumb,* en la denominació anglesa.

3.3.2.1.- Models històrics

És el mètode més utilitzat encara per a fer l'estimació de costos és basa en l'experiència personal de qui la fa (per exemple, el cap de projecte). Com apunt crític , es tracta d'un mètode marcadament subjectiu i molt sotmès a errors.

Aquest model és aplicable tant en metodologies descendents com ascendents. En les estimacions ascendents, s'avaluen cadascun de les fases del projecte per separat sumant-les al final per obtenir el total.

Combinació d'estimacions individuals

Quant és tenen més d'una font d'opinió cal combinar les diferents estimacions, per aquest cas disposem de diverses tècniques que ens poden ajudar en creació de l'estimació fina. A continuació esmentem tres d'aquestes tècniques:

a) **Mitjana**. La manera més senzilla per a arribar a una combinació adient és, simplement, calcular la mitjana entre les diverses estimacions obtingudes per separat.

b) **Mitjana ponderada**. Un altre procediment consisteix a tenir present que hi pot haver estimacions pessimistes i optimistes, i que cal tenir-les en compte d'una manera diferent a altres estimacions intermèdies. El que es fa, doncs, en lloc de calcular la mitjana habitual, és establir una mitjana ponderada, que podria ser, per exemple, en el cas de tres estimacions, la que recomanava Pressman en les primeres edicions del seu llibre E*nginyeria del Software:*

$E = (o + 4n + p) / 6,$

on *e* és l'estimació final, *o* és l'estimació més optimista, *p* la més pessimista i *n* la que ha quedat enmig.

c) **Tècniques de Delphi**. Aquest és el procediment més seriós i complex. Va ser elaborat per la Rand Corporation l'any 1948 per a evitar les reunions de grup entre estimadors, per la perillosa dinàmica que tenien. La idea és que diversos estimadors donin per separat la seva estimació després d'haver estudiat el projecte. Aquests resultats es traslladen anònimament als altres estimadors, que, d'aquesta manera, poden matisar i refer la pròpia estimació considerant el que han proposat els altres. El procés es repeteix tantes vegades com faci falta fins a aconseguir una certa convergència que sigui almenys satisfactòria als ulls del coordinador del procés. Si així no s'arriba a la convergència desitjada, el coordinador discuteix personalment amb cada estimador les diferències trobades per mirar de forçar una solució acceptable per a tothom.

3.3.2.2.- Models empírics o estadístics

Els models empírics volen aprofitar l'estudi estadístic de les dades reals disponibles mesurades en els projectes ja acabats. Es vol buscar fórmules que relacionin l'esforç amb les diverses variables que hi puguin intervenir.

Als anys seixanta, es va provar d'aproximar les dades estadístiques disponibles amb expressions lineals del tipus següent:

$$E = C_o + \sum_i c_i \cdot x_i.$$

En aquesta expressió, E és l'esforç, sovint mesurat en persones - mes, x_i són les diverses variables o els atributs que es creu que incideixen en el cost, sovint anomenades CDA, mentre que c_i són els coeficients que resulten de l'aproximació estadística a partir de les dades disponibles procedents de projectes anteriors ja acabats i ben mesurats.

Es va notar que els models lineals no eren bons per la valoració i es va abandonar aquest plantejament per a intentar trobar fórmules de tipus exponencial de la manera següent:

$$E = (a + b \cdot L^c) \cdot m(x),$$

on E és l'esforç mesurat en persones - mes, *L* és la grandària estimada del projecte en KLOC, mentre que *a, b* i *c* són els coeficients que resulten de l'aproximació estadística de les dades disponibles.

El terme addicional *m(x)*, que no tos els models utilitzen, és un element d'ajust per a tenir en compte més dades que simplement la grandària del projecte expressada en KLOC. De fet, el terme *m(x)* és aquí l'equivalent a les característiques generals que completaven i permetien ajustat els punts de funció d'Albrecht deduïts de les característiques funcionals. La idea bàsica és que l'esforç tal vegada no depèn només de la grandària (les línies de codi) i que hi ha altres factors que hi influeixen.

Gran part dels models empírics es van trobar durant els anys setanta, que mesuraven dades de projectes que no tenen res a veure en els projectes actuals, de manera que l'estimació de l'esforç es exagerat comparat amb els projectes actuals degut als avenços en aquestes àrees, on els coeficients de model són en l'actualitat força diferents.

Veiem un parell d'exemples de models estadístics que ja han esdevingut clàssics.

1) El model de Walston i Felix

El primer exemple de model estadístic és l'estudi, publicat l'any 1977, que Walston i Felix van fer a partir de seixanta projectes acabats entre 1973 i 1977. Es tractava, possiblement, dels primers projectes disponibles, no eren homogenis (hi havia diferents llenguatges de programació, diferents ordinadors i aplicacions diverses) i, a més, eren de grandàries diverses (de 12 a 11.758 mesos - home i de 460 a 467.000 LOC). Els coeficients eren *a = 0, b* = 5,2 i *c* = 0,91 i la fórmula en aquest cas és la següent:

$$E = 5{,}2 \cdot L^{0{,}91}$$

També s'obtenien altres informacions, com la duració en mesos del projecte *(D)*, el nombre de persones implicades *(P)* i el nombre de pàgines de documentació *(DOC),* gràcies a les fórmules següents:

- $D = 4{,}1 \cdot L^{0,36}$.
- $P = 0{,}54 \cdot E^{0,6}$.
- $DOC = 40 \cdot L^{1,01}$.

2) El model de Bailey i Basili

Un altre exemple clàssic és el de Bailey i Basili, publicat l'any 1981, a partir de dades de divuit projectes més homogenis, tots del Goddard Space Center de la NASA i programats en Fortran. En aquest cas, els coeficients són *a* = 5,5, *b*= 0,73 i *c* = 1,16; és a dir, s'obté l'expressió següent:

$$E = 5{,}5 + 0{,}73 \cdot L^{1,16}$$

Hauria de sorprendre el fet que l'exponent de les KLOC sigui inferior a la unitat en el cas de Walston i Felix, mentre que és clarament superior a la unitat en el cas de Bailey i Basili. La raó és que, en l'estudi de Walston i Felix, entre les LOC es comptava fins a un màxim del 50% de comentaris, mentre que a l'estudi de Bailey i Basili, com a la gran majoria dels models estadístics, no s'inclouen els comentaris en el recompte de les LOC.

També ha una gran varietat de fórmules obtingudes a partir de dades procedents de diversos projectes, entre elles algunes que relacionen l'esforç amb els punts de funció d'Albrecht[20].

Cal tenir en compte que els resultats d'esforç que s'obtenen en els diversos models no tenen pas per què coincidir, ja que reflecteixen dades extretes de projectes diferents i no sempre comparables ni homogenis. De fet, el models estadístics esmentats es consideren ara obsolets del tot i proporcionen tan
sols un marge altíssim per a una estimació raonable. Models posteriors, com el COCOMO de Boehm, han millorat i actualitzat les dades i les fórmules.

3.3.2.3.- Models teòrics: el model de Putman

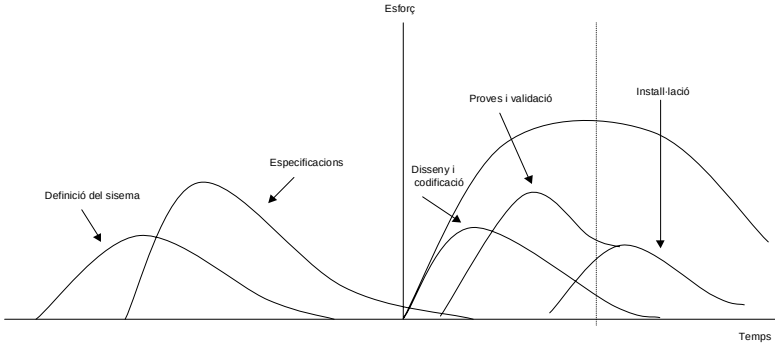
Al 1978, Lawrence H. Putman[21] va començar a plantejar la necessitat de models teòrics que, prescindint de la possibilitat de disposar o no de dades de projectes ja acabats, tinguessin en compte la manera com es distribueix l'esforç.

Putman va adoptar la corba de Rayleigh/Norden que es mostra a la figura següent per la distribució d'esforços en el temps:

[20] El model d'Albrecht i Gaffney és un exemple de model que relaciona l'esforç amb els punts de funció d'Albrecht. Aquest model proposa la fórmula següent: $E = -13{,}39 + 0{,}054\ FP$.

[21] L.H. Putman (1978). "A General Empirical Solution to the Macro Software Sizing and Estimation Problem". *IEEE Transactions on Software Engineering* (vol. SE-4, núm. 4, juliol, pàg. 345-361).

Corba de Rayleigh/Norden de l'esforç de desenvolupament



L'estudi de la corba permet, entre altres coses, relacionar l'esforç (*K*, en persones - any), les línies de codi (*L*, en KLOC) i el temps total de desenvolupament del projecte (t_D en mesos) segons la fórmula:

$$L = C_k \cdot (K^{(1/3)} \cdot t_D{}^{(4/3)}.$$

La idea és que, a partir d'un esforç (*K*) i un temps de desenvolupament (t_D), la fórmula indica la productivitat en la construcció de programari i ens dóna el nombre de KLOC que s'obtenen.

La fórmula inclou una constant anomenada **constant tecnològica** (C_k), que, en paraules de Putman, és en certa manera una mesura de l'estat de la tecnologia que s'aplica al projecte. La constant C_k reflecteix l'efecte en la productivitat d'una munió de factors com el maquinari i les seves limitacions, la complexitat dels programes, l'experiència del personal (tant tècnics com usuaris), l'entorn de desenvolupament i programació, etc. Això és el que fa que el mètode teòric de Putman no esdevingui mai obsolet. Es tracta d'anar canviant els valors de la constant tecnològica a mesura que canvien els entorns de desenvolupament i programació. El problema és que conèixer el valor correcte de C_k no és pas senzill.

Al començament, Putman indicava una sèrie discreta de vint valors de C_k que variaven entre 610 i 57.314. La C_k doncs, varia d'un valor prou baix, 610, fins a un altre que és quasi cent vegades més alt. És a dir, el paper d'aquesta constant és del tot fonamental en el model.

L'any 1981, Putman presentava per primera vegada el seu model inserit en una eina automatitzada anomenada **SLIM**, que, a partir de diverses preguntes, determina la constant tecnològica corresponent.

El model de Putman, en la seva darrera versió, identifica la fórmula anterior com l'anomenada **equació del programari** i, a més, descompon la constant tecnològica C_k en dues noves constants:

$C_k = P/B^{0,333}$,

on *P* és el **paràmetre de productivitat** (que seria de 28.000 en el cas d'aplicacions de la informàtica de gestió), i *B* és un **factor especial d'habilitat** que s'incrementa lentament a mesura que creixen la necessitat d'integració, les proves, la garantia de qualitat,

la documentació i altres exigències que disminueixen la productivitat.

En la pràctica, avui el model de Putman només pot ser ben utilitzat per aquells que adquireixen l'SLIM, L'SLIM és capaç, després de un munt de preguntes, de determinar C_k i la resta de dades que interessen per a l'estimació d'un projecte.

Les conclusions a què s'arriba en la valoració del model és que el model és prou correcte per a projectes molt grans i que el resultat és molt més imprecís i dubtós en el cas de projectes petits i mitjans.

És indispensable disposar de l'eina informàtica SLIM per determinar la constant tecnològica (C_k) al model de Putman de gran efecte en el resultat final.

3.3.2.4.- Models compostos: els models COCOMO de Barry W. Boehm

El model va ser creat per Barry W. Boehm i es va publicar per primera vegada al llibre de 1981 *Software Engineering Economics*. Després ha tingut diverses versions, com l'Ada i, sobretot, l'actual projecte COCOMO II, en què Boehm col·labora en el desenvolupament al Center for Software Engineering (CSE) de la Universitat del Sud de Califòrnia (University of Southern California, USC).

Diverses versions del programari tenen objectius diferents i que la darrera versió (COCOMO II) no substitueix pas les anteriors (COCOMO 81 i Ada COCOMO), que són vàlides sempre que es facin servir els paradigmes de construcció de programari i el cicle de vida i les eines per als quals es van definir.

Podem trobar moltes implementacions informatitzades com, per exemple, el COSTAR (*Software Cost Estimation Tool)*, una eina ja clàssica de l'empresa Softstar Systems, que, a partir del COSTAR 5.0, implementa ja el COCOMO II.

El COCOMO clàssic de 1981

Format per, tres models diferents, que tenen en compte diversos graus de complexitat:

1) El **COCOMO bàsic** és un model estàtic vàlid per a obtenir una estimació ràpida de l'esforç (mesos-home) en funció de la grandària (KLOC) a l'inici del cicle de vida.

2) El **COCOMO intermedi** afegeix al càlcul de l'esforç en funció de la grandària, l'efecte d'uns atributs que influeixen en el cost (CDA), amb els quals es vol tenir en compte el tipus d'aplicació i tecnologia, les qualificacions i l'experiència del personal, l'entorn de disseny i programació i les eines de què disposa, etc.

3) El **COCOMO avançat** incorpora totes les característiques de la versió intermèdia, però en lloc d'avaluar els CDA amb un únic valor per a tot el cicle de vida, té en compte diversos CDA per a cada fase[22] de la construcció del programari.

[22] L'anàlisi, el disseny, la programació, les proves, etc.

Les equacions del model COCOMO tenen sempre la forma general que mostrem a continuació:

- $E = a \cdot L^b \cdot CDA$,
- $T = c \cdot E^d$,

on *E* és l'esforç (en persones - mes), *L* són les línies de codi (en KLOC), *T* és el temps de desenvolupament del projecte (en mesos) i CDA els *cost driven attributes* que, recordem-ho, no s'utilitzen en el model bàsic. Finalment, *a, b, c* i *d* són coeficients que el model proporciona com a resultat d'analitzar les dades de seixanta-tres projectes fets entre 1965 i 1980, escrits en cinc llenguatges diferents, de grandàries que varien de 2 a 1.000 KLOC, i amb productivitats que varien de 28 a 250 LOC per persona - mes.

COCOMO té en compte diversos tipus de projecte. Segons Boehm, els tres tipus de projecte són els que presentem tot seguit:

a) **Orgànic** (*organic mode).* Projectes relativament petits i senzills, amb poca innovació tecnològica, on treballa un equip petit format per gent que té força experiència en el tipus d'aplicació concret i que, a més, ha de tenir requeriments poc rígids.

b) **Semiacoblat** (*semidetached).* Projectes de nivell intermedi en grandària, complexitat i sofisticació tècnica, on diversos equips diferents col·laboren per a construir una aplicació amb requeriments mitjanament rígids.

c) **Encastat** (*embedded mode*). Projectes d'una grandària i complexitat francament elevada, on intervenen diversos equips diferents i els requeriments, tant de maquinari com de programari, són molt rígids.

Cal tenir present que, en la informàtica de gestió, possiblement només tinguin sentit els projectes orgànics o semiacoblats i que, tal com diu Pressman, els projectes encastats suggereixen una altra mena de sistemes informàtics amb requeriments molt rígids[23].

La següent taula recull els coeficients per al model bàsic.

Coeficients del model COCOMO bàsic				
Projecte	Coeficients			
	a	*B*	*c*	*D*
Orgànic	2,4	1,05	2,5	0,38
Semiacoblat	3,0	1,12	2,5	0,35
Encastat	3,6	1,20	2,5	0,32

En el model intermedi els coeficients són els d'aquesta altra taula:

Coeficients del model COCOMO intermedi				
Projecte	Coeficients			
	a	*b*	*c*	*d*
Orgànic	3,2	1,05	2,5	0,38
Semiacoblat	3,0	1,12	2,5	0,35
Encastat	2,8	1,20	2,5	0,32

[23] Com, per exemple, el sistema de control de navegació d'un avió.

Hi ha quinze paràmetres que recullen les característiques del producte (CDA) que s'ha de fer, de l'ordinador amb què es treballa i que executarà l'aplicació, del personal que forma l'equip del projecte i del projecte mateix. Cadascun d'aquests quinze paràmetres pot agafar diversos valors: molt baix, baix, nominal, alt, molt alt i extra-alt[24] que varien al voltant de la unitat. La taula següent mostra els valors dels CDA:

COCOMO intermedi: atributs que influeixen en el cost (CDA)							
		Valors					
		Molt baix	Baix	Nominal	Alt	Molt alt	Extra-alt
Atributs del producte							
RELY	Fiabilitat requerida	0,75	0,88	1,00	1,15	1,40	-
DATA	Volum de la base de dades	-	0,94	1,00	1,08	1,16	-
CPLX	Complexitat	0,70	0,85	1,00	1,15	1,30	1,65
Atributs de l'ordinador							
TIME	Limitacions de temps d'execució	-	-	1,00	1,11	1,30	1,65
STOR	Limitacions de volum de memòria	-	-	1,00	1,06	1,21	-
VIRT	Volatilitat de la màquina virtual	-	0,87	1,00	1,15	1,30	-
TURN	Temps de resposta	-	0,87	1,00	1,07	1,15	1,65
Atributs del personal							
ACAP	Capacitat d'anàlisi	1,46	1,19	1,00	0,86	0,71	1,65
AEXP	Experiència en l'aplicació	1,29	1,13	1,00	0,91	0,82	1,56
PCAP	Capacitat de programació	1,42	1,17	1,00	0,86	0,70	-
VEXP	Experiència en la màquina virtual	1,21	1,10	1,00	0,90	-	-
LEXP	Experiència en els llenguatges de programació	1,14	1,07	1,00	0,95	-	-
Atributs del projecte							
MODP	Ús de pràctiques modernes	1,24	1,10	1,00	0,91	0,82	-
TOOL	Ús d'eines de programari	1,24	1,10	1,00	0,91	0,83	-
SCED	Exigències de planificació	1,24	1,08	1,00	-	1,10	-

Algunes hipòtesis del model COCOMO

Algunes de les hipòtesis que fa servir el model COCOMO són, entre d'altres, les següents:

- No es tenen en compte els comentaris en el recompte de les KLOC.

[24] *Very low, low, nominal, high, very high, i extra high.*

- S'admet l'equivalència següent: 1 mes - home són 152 hores de feina.
- Es considera que els requeriments són estables.
- S'accepta que el projecte està ben gestionat.

El COCOMO II dels anys 90 i els 2000

El COCOMO II, que encara està en fase d'elaboració i de prova es centra en l'estimació de costos i planificació del programari especialment adient per als cicles de vida utilitzats en els anys 90 i a partir del 2000.

COCOMO II inclou tres models que corresponen a diferents fases i modalitats del futur cicle de vida:

1) **Model de composició d'aplicacions** (*Application Composition Model):* inclou l'ús de prototips per a disminuir els riscos potencials que sorgeixen amb les interfícies gràfiques d'usuari típiques d'eines RAD i altres eines actuals de productivitat i de l'orientació a objectes. En aquest model es defineixen uns punts objecte (*object point)* que vindrien a ser una adaptació i modernització dels punts de funció d'Albrecht ja vistos.

2) **Model de disseny primerenc** (*Early Design Model*): intenta obtenir una primera aproximació en les fases inicials del cicle de vida, quan encara es coneixen poques de les característiques i dades definitives del projecte. Utilitza com a primitives de sortida tant les línies de codi com els clàssics punts de funció d'Albrecht sense ajustar (TUFP).

3) **Model de post-arquitectura** (*Post-Arquitecture Model)*: versió més completa, que correspon a la modernització del COCOMO tradicional de 1981. S'aplica quan es considera que el projecte disposa ja de requeriments estables. D'altra banda, també fa servir com a primitives de sortida les línies de codi i els punts de funció d'Albrecht sense ajustar (TUFP) i, a més a més, té en compte indicadors de la reutilització de programari, cinc factors d'escala i fins a disset factors específics diferents.

És important destacar com en el projecte COCOMO II, en el model post - arquitectura, es poden fer servir les noves línies de codi (NSLOC[25]) quan es tracta d'estimar els costos d'una aplicació nova, o les línies de codi adaptades (ASLOC[26]) quan es tracta d'estimar una aplicació en la qual s'utilitzarà bàsicament el
disseny o el codi d'altres aplicacions ja disponibles. És a dir, el COCOMO II inclou ja com a aspecte central el fenomen de la reutilització del programari.

L'ús d'estàndards

Per poder tenir-los cal recollir dades de projectes anteriors de la mateixa instal·lació i establir quin és l'esforç que correspon a les diferents activitats

3.3.2.5.- L'ús d'estàndards de productivitat

Mai no és fàcil determinar quin és model que ens ajudarà per l'estimació de costos ni quin serà el que millor es pugui ajustar al nostre projecte.

[25] *New Source Lines of Code*
[26] *Adapted Source Lines of Code*

La pràctica professional de cada dia ha causat que moltes instal·lacions[27], tot i tenir com a referència les dades que ofereixen els models, hagin decidit adoptar un sistema molt més simple i menys complicat: elaborar estàndards de productivitat propis.

En aquest tipus de model la idea és quantificar mitjançant projectes anteriors l'esforç extraient d'aquest projectes anteriors el atributs que puguin ser quantificats.

En lloc de quantificar cada activitat o conjunt d'aquestes característiques funcionals en línies de codi o punts de funció i buscar models que converteixin aquestes mètriques en esforç (mesos - home), n'hi ha prou de disposar directament, per a cada activitat, de l'esforç estàndard que ha requerit en altres projectes anteriors.

També, al marge dels estàndards propis de cada instal·lació, a la bibliografia tècnica[28] es troben les **recomanacions pràctiques a cop d'ull**.

Tal com recorda Caper T. Jones[29] en un article, les recomanacions pràctiques a cop d'ull segueixen essent populars, però mai no són exactes i no substitueixen els mètodes formals d'estimació del programari.

Segons Jones, es donen les equivalències pràctiques següents:

- 1 FP = 100 LOC
- FP eleva a 0,4 = mesos de desenvolupament.
- FP/150 = nombre de persones que calen per al desenvolupament.
- FP/500 = nombre de persones que calen per al manteniment futur.
- FP elevat a 1,15 = nombre de pàgines de documentació.
- FP elevat a 1,2 = nombre de casos de prova que es fan.
- FP elevat a 1,25 = potencial d'errors (en projectes nous).
- FP elevat a 0,25 = nombre d'anys que seguirà en ús l'aplicació.

Les recomanacions pràctiques a cop d'ull poden ser més generals, com les que, d'acord amb Jones, són:

- Si no es vigilen els requeriments creixents, els usuaris aconseguiran que augmentin l'1% cada mes de desenvolupament del projecte: en un projecte de dos anys hi ha, al final, el 24% més de requeriments dels que hi havia al començament.

- corregeix el 30% dels errors que hi ha.
- Els projectes inverteixen aproximadament el 18%

Ús d'estàndards de productivitat

És imprescindible la utilització d'estàndards de productivitat quan, per exemple, s'utilitza l'*outsourcing* (externalització) i s'encarrega la programació d'una nova aplicació a una empresa externa. En aquest cas, cal tenir dades clares que permetin a les dues empreses (la que vol el projecte i la que ha de fer la programació) posar-se d'acord en el cost que s'ha de pagar per la feina de programació que cal fer.

[27] Generalment, les instal·lacions més grans i amb més experiència.
[28] R.O. Grady (1982), *Practical Software Metrics for Project Management and Process Improvement* (cap. 2, pàg. 13-20). Englewood Cliffs: Prentice Hall (Hewlett Packard professional Books).
[29] C.T. Jones (1996) "Software Estimating Rules of Thumb". *IEEE Computer* (març, pàg. 116-118).

- corregeix el 30% dels errors que hi ha.
- Els projectes inverteixen aproximadament el 18% del seu esforç total a determinar els requeriments i fer les especificacions, un 19% en el disseny, un 34% en la codificació i un 29% en les proves.
- Es gasten dues o tres vegades més d'esforç a mantenir i millorar el programari que el que es gasta a crear-lo.
- Hi ha aproximadament un defecte sense detectar per cada deu que se'n troben a les proves abans de distribuir una versió del programari.

3.3.3.- Aplicació dels mètodes i models d'estimació en la gestió d'altres tipus de projectes informàtics

A diferencia de les mètriques no tots els models especificats ens són vàlids per la gestió de la resta de projectes, si més no n'hi ha alguns models que ens poden ajudar com per exemple els models històrics o en cert casos els estadístics sempre que tinguem alguna dada mesurable.

Una idea que si es força genèrica i ens pot ajudar són els mètodes definits per Barry W. Boehm on ens diu que els podem definir en set categories:

- Utilització de models: models algorísmics
- Judici d'experts en el tipus de projecte
- Analogia amb un altre projecte semblant
- Utilitzar tots els recursos disponibles
- Preu guanyador
- Estimació global descendent
- Descomposició en activitats (WBS) i estimació ascendent

Aquesta definició de mètodes també ens pot ajudar en la avaluació i gestió de qualsevol tipus de projecte, encara que el mètode més utilitzat es potser la Descomposició en activitats (WBS) i estimació ascendent degut a que en l'actualitat un projecte és el conjunt divisible d'altres més petit i fàcils de gestionar.

4.- Punts generals per l'estimació d'esforç en la gestió de projectes.

El mon de la informàtica, és un mon dinàmic, sempre subjecte a canvis, l'evolució fa que mai puguem definir un estàndard per la catalogació de projectes i que les pautes que utilitzem avui siguin valides en un futur.

Aquest efecte canviant fa que els diferents models o mètodes que podem utilitzar sempre hagin de dependre de definicions molt genèriques i que els seves mètriques hagin de ser tornades a calcular per una utilització òptima.

Partint una mica de visions genèriques és del tot lícit dir que:

En tot projecte es podran definir unes etapes:

La idea de que tot projecte pot ser separat en diferents tasques ens permet pensar que podem buscar etapes per la posterior valoració. Com ja em vist les diferents etapes està perfectament definides de manera genèrica i són : Inici del projecte, qualificació del projecte, desenvolupament del projecte i tancament del projecte.

Tot projecte tindrà elements que permetin mesurar-lo.

Els elements que escollim per valorar el projecte han de ser presents durant totes les etapes i a més a més han de ser mesurables per poder garantir la qualitat, el volum i la productivitat del projecte.

Mentre que en projectes de creació de programari podem valorar mètriques com la quantitat de línies en altre tipus de projecte podem valorar per exemple quantitat de usuaris afectats.

Tot projecte podrà ser valorar utilitzant altres projectes semblants acabats.

Una de les vies més ràpides per la valoració és tenir exemples, abans de valorar qualsevol projecte es important documentar-se per esbrinar si som els primers o com és normal ja existeixin projectes com el que nosaltres volem fer totalment acabats. D'aquesta manera podem intentar minimitzar els problemes que han tingut en la gestió del nostre.

5.- Conclusions

Inicialment la idea de buscar les mètriques i els models aplicables a gestió de projecte venia donat pel fet de que en l'actualitat cada vegada està agafant amb més força la necessitat de quantificar l'esforç dels projectes abans, durant i després de la seva execució.

El camí informàtic que em d'utilitzar a l'hora de valorar qualsevol projecte és la seva descomposició en tasques més petites que ens permetin fe valoracions més acurades on les mètriques estan molt més definides i constants durant tot el procés.

S'ha definit el concepte mètrica i s'han valorat diferents tipus, quedant pales que cada projecte o part del projecte està delimitat per les mètriques que el defineixen com atributs, per exemple una mètrica constant a tots els projectes és la utilització de recursos humans on s'utilitza la definició persona-mes per poder valorar l'esforç necessari. S'ha de tenir present que tots els projectes no poden ser mesurats per les mateixes mètriques i que s'ha d'estudiar el projecte per esbrinar les mètriques que el defineixen.

Una vegada definides les mètriques podem utilitzar els diferents models existent en l'actualitat per fer l'estimació definitiva, al igual que en les mètriques no tots els models són aplicables a tots els projectes, es més en alguns casos només podem utilitzar models històrics, es a dir només experiències anteriors ens són útils per la gestió del projecte.

Si més no una conclusió general a tot el tema és que l'existència de models i mètriques de mesura està molt desenvolupat en el àmbit del projectes d'enginyeria de programari. Aquest fet pot venir de que històricament la programació ha estat el nucli central de la informàtica i ha format part d'un gran nombre dels projectes desenvolupats.

En l'actualitat molts dels projectes que es duen a terme no són de programació, els projectes de comunicacions, infra estructura, etc. estan assolint major prioritat que la creació de programari a mida.

El problema que ens trobem amb el nou gir que està fent el mercat informàtic és que un responsable de projecte ha de poder tenir eines que el puguin ajudar a l'hora de valorar aquest nous tipus de projectes al igual que ho fa amb els projectes de creació de programari. Actualment és un mercat obert on hi ha empreses que treballen per assolir un software capaç de donar servei en aquest tipus de necessitats.

El caràcter fluctuant de la informàtica ens obliga a esbrinar continuadament nous mètodes per l'estimació d'esforç basats moltes vegades en la pròpia experiència d'altres projectes de menor grandària però d'imatge semblant, d'aquesta manera és tendeix sovint a un mètode ajustat de valoració.

6.-Glossari

CDA : Les variables dels models empírics que poden incidir amb el cost

COCOMO: model algorísmic d'estimació de costos molt conegut i utilitzat, desenvolupat per W.Boehm a partir de 1981.

COCOMO II: Revisió de COCOMO

DET: nombre d'element, o camps de dades, d'un fitxer.

DSI: instruccions de codi font lliurades realment.

EI: entrades externes que fan referència als tractaments que processen dades o informació de control que s'introdueix a l'aplicació des de fora dels seus límits

EIF: fitxers d'interfícies externes

EO: sortides externes

EQ: consultes externes

Estàndards de productivitat: dades pròpies d'una instal·lació sobre la productivitat en la construcció de programari, obtingudes a partir de projectes anteriors, tipificant-ne els resultats

FP: punts de fusió

Home-mes o persona-mes: unitat de mesura que representa un mes de feina d'una persona de l'equip de professionals que treballa en un projecte informàtic.

IFPUG: grup internacional d'usuaris dels punt de fusió

Indicador de programari: combinació de diverses mètriques que proporciona una visió resumida del producte o del procés de construcció de programari.

KDSI: milers de DSI

KLOC: milers de línies de codi

LIF: fitxers lògics interns

Línies de codi : unitat de mesura que inclou en el recompte, a més de les línies codificades, les directives de compilació, les declaracions de les estructures de dades i el codi executable; no es compten els comentaris i les línies generades automàticament per l'entorn de programació o que són fruit de reutilització.

LOC: línies de codi

Mètrica de programari: assignació de valors a un atribut d'una entitat pròpia del programari, ja sigui un producte o un procés

Mètriques de productivitat: mètriques que recullen l'eficiència del procés de producció de programari i relacionen el programari construït amb l'esforç que ha costat obtenir-lo.

NCSS: línies de codi font que no tenen en compte els comentaris

NSLOC: noves línies de codi font

OBS: Organized Breakdown Structure. Descomposició de l'organització per a atendre totes les tasques del projecte.

PBS: Product Breakdown Structure. Descomposició en parts del producte final: taules de la base de dades, formularis, programes, mòduls, transaccions, etc.

PCA: ajust per a la complexitat del procés.

Punts de funció: mètrica creada per Albrecht al 1979 que pretén recollir les funcionalitats com a mesura de la dificultat de construir un programari. El mètode de càlcul fa el recompte d'unes característiques funcionals, tenint en compte la complexitat i, després, corregeix el valor obtingut mitjançant l'estimació de fins a catorze característiques generals que determinen l'entorn de treball i del projecte.

RAD: Rapid Aplication Development . Desenvolupament ràpid d'aplicacions que permeten noves eines.

RET: nombre de registres elementals diferents.

TDI: grau total d'influència.

TUFP: total de punts de funció sense ajustar

WBS: descomposició estructural dels treballs que cal fer, es a dir, la llista estructurada de totes les activitats i tasques d'un projecte.

7.-Bibliografia.

Gestió d'una organització informàtica
(mòdul 4 El projecte informàtic de construcció de programari, mòdul 5 Estimació de costos d'un projecte informàtic)
Miquel Barceló García , Joan Antoni Pastor i Collado (setembre 1999 Universitat Oberta de Catalunya)

Project Management for Construction Fundamental Concepts for Owners, Engineers, Architects and Builders
by Chris Hendrickson, Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA l52l3 Copyright C. Hendrickson 1998
First Edition originally printed by Prentice Hall, ISBN 0-13-731266-0, 1989 with co-author Tung Au. Second Edition prepared for world wide web publication in 2000.
http://www.ce.cmu.edu/pmbook/index.html

A Guide to the Project Management Body of Knowledge PMBOK ® **Guide** 2000 Edition **PROJECT MANAGEMENT** INSTITUTE PMBOK® **Guide** - 2000 Edition

http://www.pmi.org/prod/groups/public/documents/info/pp_pmbokguide2000excerpts.pdf

Proyecto Métrica Versión 3. Metodología de planificación y desarrollo de Sistemas de Información*, 2000*
MAP (Ministerio para la Administraciones Públicas). Consejo Superior de Informática,

http://www.csi.map.es/csi/metrica3/index.html

Gestió dels sistemes d'informació a l'empresa
Àngel Ros Domingo, Jordi Viñallonga Plaza Data de publicació: 1995 Col·lecció: Aula Teòrica, 36

http://bibliotecnica.upc.es/edupc/locate4.asp?codi=IN002XXX

Simulation and Comparison of Albrecht's Function Point and DeMarco's Function Bang Metrics in a CASE Environment
R. Rask, P. Laamanen, K. Lyyttinen

http://csdl.computer.org/dl/trans/ts/1993/07/e0661.pdf

Software Quality Metrics Overview
By Stephen H. Kan.

http://www.awprofessional.com/articles/article.asp?p=30306&seqNum=1

Metrics and Models in Software Quality Engineering, 2nd Edition
By Stephen H. Kan.
Published by Addison Wesley Professional.

http://www.awprofessional.com/title/0201729156#

Verifying function point values
E.E. Rudolph, HS Bremerhaven, erudolph@hs-bremerhaven.de G.E. Wittig, Bond University, gerhard_wittig@bond.edu.au G.R. Finnie, Bond University, gavin_finnie@bond.edu.au P.M. Morris, Total Metrics, Pam.Morris@Totalmetrics.com

http://www.totalmetrics.com/tmdl/R022%20AHP%20Method%20of%20Verifying%20Function%20Point%20Values.pdf

Basic COCOMO

Cost Estimating Web Site
http://www.jsc.nasa.gov/bu2/COCOMO.html

COCOMO II Research Project
http://sunset.usc.edu/research/COCOMOII/index.html

IFPUG - International Function Point Users Group - Home Page
http://www.ifpug.org/

Function Point FAQ
http://ourworld.compuserve.com/homepages/softcomp/fpfaq.htm

Measuring Application Development Productivity (pàg. 83-92)
Prooceeding Joint SHARE/GUIDE/IBM Applications Development Symposium (1979) Albrecht A.J.

Software Engineering Economics
Englewood Cliffs: Prentice Hall
Boehm, B.W.

Object-Oriented Project Manager with UML
John Wiley & Sons
Cantor M.R.

Controlling Software Projects
Englewood Cliffs:Prentice Hall
De Marco, T.

A general empirical solution to the Marco software sizing and estimation problem
IEEE Transaction on software Engineering
Putman LH

http://www.computer.org/publications/dlib/

A method of programing measurement and estimation
IBM Systems journal (num. 1 pàg. 14-73)
Walston C.E., Felix C.P.

8.- Annexos

Alguns venedors de software d'estimació de costos

Galorath Associates, Inc.
GA Seer Technologies Division
PO Box 90579
Los Angeles, CA 90009
Voice: (310) 670-3404; FAX: (310) 670-6481
E-mail: info@galorath.com
URL: www.galorath.com

Lockheed-Martin Price Systems
700 East Gate Drive - Suite 200
Mount Laurel, NJ 08054
Voice: (800) 437-7423; FAX: (609) 866-6789
E-mail: Bruce.Fad@DEN.MMC.Com

Quantitative Software Management, Inc.
President: Lawrence Putnam, Sr.
2000 Corporate Ridge - Suite 900
McLean, VA 22102
Voice: (800) 424-6755; FAX: (703) 749-3795
E-mail: 76274.72@compuserve.com
URL: www.qsm.com

Rubin Systems, Inc.
5 Winterbottom Lane
Pound Ridge, NY 10576
Voice: (914) 764-4931; FAX: (914) 764-0536
E-mail: 71031.377@compserve.com

Softstar Systems
Dan Ligett
P.O. Box 1360
Amherst, NH 03031
Voice: (603) 672-0987; FAX: (603) 672-3460
E-mail: info@softstarsystems.com
URL: www.SoftstarSystems.com
Software Productivity Research
Chairman: Capers Jones
One New England Executive Park
Burlington, MA 01803-5005
Voice: (617) 273-0140; FAX: (617) 273-5176
E-mail: barbara@spr.com
URL: www.spr.com/