1.

El mite home-mes consisteix en un seguit d'idees promogudes per Fred Brooks. Basant-se en la seva experiència sent l'administrador del projecte OS/360, va arribar a la següent conclusió:

- **Afegir recursos humans a un projecte amb retard, només pot provocar un endarreriment superior a l'actual.** És a dir, com la principal causa dels problemes en el desenvolupament software està en la planificació i distribució de recursos, així com que les hores i els recursos humans són intercanviables, fa que la direcció del projecte prengui decisions equivocades com la de portar més empleats (més mà d'obra).

Ara ve la següent qüestió: *Com és pot endarrerir un projecte amb més mà d'obra?*

- La resposta és ben senzilla: Un projecte ja començat té una planificació d'hores i de personal concret, amb el qual la feina està definida. Mà d'obra nova significa que aquesta persona ha de reunir-se per l'assignació de tasques i objectius, formar-se per portar les tasques a terme, introduir-lo al projecte re-assignant noves tasques al personal empleat,...

Per evitar aquestes situacions, abans de començar un nou projecte, s'ha de fer una bona planificació de tasques i hores. Ens hem d'anticipar als errors. La comunicació entre els empleats ha de ser fluida i s'ha de crear un bon ambient de treball per a què hi hagi motivació.

2.

La millor opció en aquest cas és utilitzar el desenvolupament de programari Scrum. Aquest desenvolupament permet la iteració i incrementació. Això vol dir que el client pot participar per a que pugui guiar de forma regular els resultats del projecte. Amb aquest fet es permet fer entregues curtes i regulars del producte final (cada dues o quatre setmanes) i aproximar-se cada cop més a les expectatives del client.

A mesura que el projecte avança, equip projecte - client cerquen més acords i s'aproximen cada cop al producte final. Cada iteració fa que l'equip del projecte millori en les seves tasques i en la seva planificació, a més, es fan reunions diàries per poder realitzar les adaptacions necessàries entre ells per a que puguin arribar a l'objectiu final de cada iteració.

La diferència de l'Scrum i el desenvolupament de programari en cascada és que el client no té un paper tant rellevant, és per això que el producte final pot no tenir tota l'expectativa que s'espera. Això és pel fet que el client no veu el producte final fins que finalitza el projecte, ja que aleshores només veu l'anàlisi mitjançant documents.

He decidit escollir Scrum ja que el client està disponible a mode de consulta, en canvi en el eXtreme Programming el client té un paper més rellevant ja que l'equip de desenvolupament segueix estrictament l'ordre de prioritat de les tasques definides pel client.

3.

L'equip de projecte estaria format pel cap de projecte i 6 empleats més. Aquests es distribueixen de la següent forma:

Cap de projecte – El cap de projecte té una remuneració de 60 €/hora. (15% temps total)

1 consultor – El consultor té una remuneració de 50 €/hora. (5% temps total)

2 analistes – Cada analista té una remuneració de 45 €/hora. (20% temps total)

3 programadors – Cada programador té una mitja (depenent si és semi sènior o sènior) de remuneració de 35 €/hora. (60% temps total)

- Càlcul hores de cada especialitat:
  - Cap de projecte: 6000h * 0.15 = 900 hores
  - Consultor: 6000h * 0.05 = 300 hores
  - Analistes: 6000h * 0.30 = 1800 hores
  - Programadors: 6000h * 0.50 = 3000 hores
- Càlcul salari total per cada especialitat:
  - Cap de projecte: 900h * 60€ = 54000 €
  - Consultor: 300h * 50€ = 15000 €
  - Analistes: 1800h * 45€ = 81000 €
  - Programadors: 3000h * 35€ = 105000 €
- Càlcul per persona:
  - Cap de projecte (100%*): 54000 €
  - Consultor (100%): 15000 €
  - Analista 1 (50%): 40500 €
  - Analista 2 (50%): 40500 €
  - Programador Semi Sènior** 1 (32%): 33600 €
  - Programador Semi Sènior 2 (32%): 33600 €
  - Programador Sènior (36%): 37800 €

    *Percentatge de les hores totals que fa cadascun del projecte per especialitat.

    ** La diferència entre un programador semi sènior i un programador sènior és que el primer té de 2 a 6 anys d'experiència i el programador sènior més de 6 anys.

- Cost total del projecte: **255.000 €**

4.

Amb la poca informació que tenim sobre l'empresa, destacaria el fet que és industrial. Si és del sector secundari, vol dir que fan un determinat o determinats tipus de productes que deriven o poden derivar a diferents tipus de models. Per aquest fet escolliria fer un software específic. No obstant, la grandària de l'empresa fa que un ERP pugui ser el més adient, ja que permet la interacció dels diferents departaments i també, amb una bon coneixement de l'empresa i una bona implantació, pot arribar a cobrir totes les necessitats, tal i com faria una software específic. A més, un ERP pot ajudar a la presa de decisions basant-se en les dades, sempre que porti incorporat eines d'ajuda a les preses de decisions.

Conclusió: en aquest cas un ERP és millor opció que un software específic.

Per a cadascun dels dos existeixen diversos riscos. El risc que tenen en comú és la implantació del sistema. Hi ha la possibilitat de que amb un ERP mai es pugui acabar de definir correctament les necessitats que té la empresa, això pot ser causa de una planificació dolenta i de no conèixer els veritables requisits de l'empresa. Amb la creació de un software a mida pot passar que una mala planificació, disseny i/o implementació faci que després de tenir el producte finalitzat, el software no sigui el que l'empresa realment requeria. Tot i així, en un producte nou sempre hi haurà una gran part de correccions d'errors on una gran part es dedicarà al manteniment.

5.

En general, un ERP es pot implantar a qualsevol tipus d'empreses. Això té dos significats: el primer és que ja sigui una empresa de transport o una universitat (dos institucions que no tenen la mateixa activitat), es pot implantar un ERP; i el segon és que un ERP és una eina molt general quan no està implantada a cap tipus de feina.

Aquesta generalització comporta la necessitat de una bona planificació, disseny i coneixement de l'activitat per tal de que l'ERP s'implanti de forma que cobreixi les necessitats de l'empresa. Amb aquest fet arribem a la conclusió que un ERP és dificultós a l'hora de la seva implantació. I no tant sols això, sino que pot ser un camí llarg. Normalment, la implantació té una durada d'1 a 3 anys, tot i així no es pot descartar que aquesta estadística es pugui trencar en un projecte concret.

Per a què una empresa utilitzi un ERP, és necessari que el treballadors tinguin una formació per saber com funciona el programa. Això té un cost força alt per l'empresa.

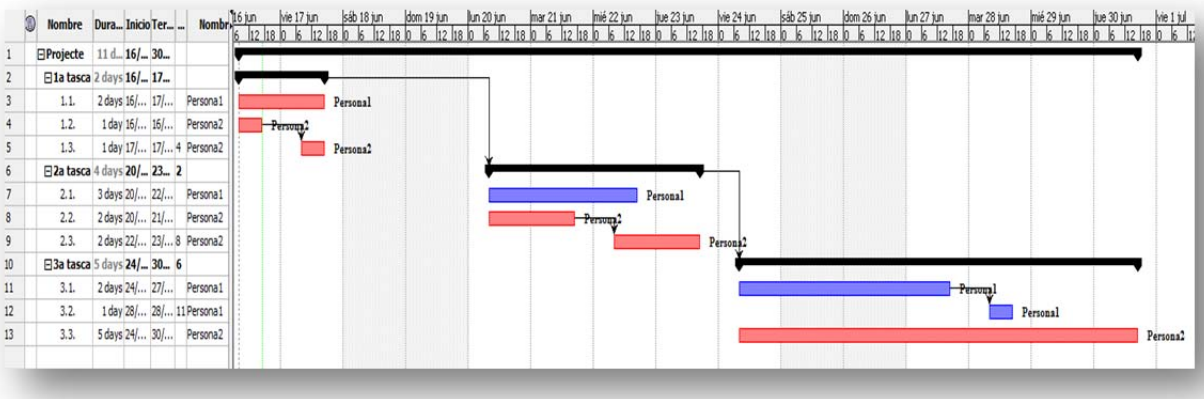
En la següent desavantatge distingim entre un ERP de programari lliure i un de privatiu. Normalment, en el privatiu, al comprar la llicència et garanteix un manteniment extern del programa, en canvi, si és lliure, l'empresa ha d'enfrontar les incidències que pugui tenir sense ajuda d'una empresa externa.

6.

Per a realitzar els diferents diagrames (Gantt i Pert) he fet servir el programa OPENPROJ, tal i com és va utilitzar en les pràctiques de l'assignatura.

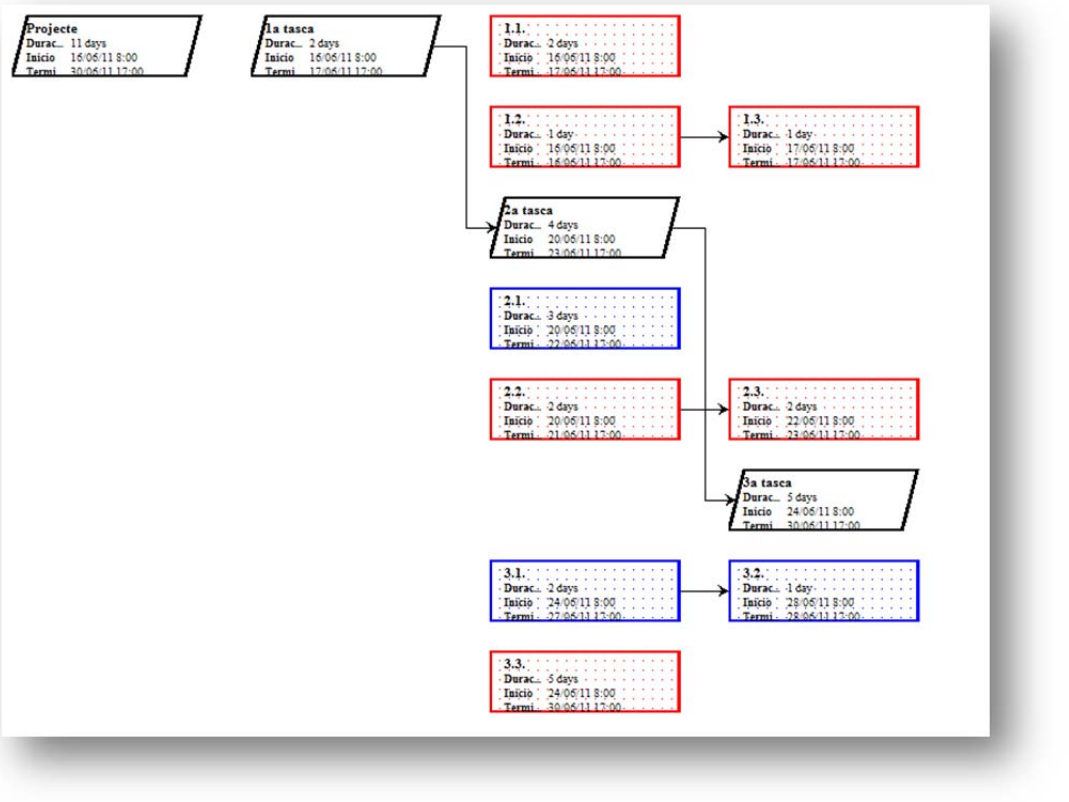
El projecte té una duració total d'onze dies.

Per començar cada tasca és necessari haver finalitzat l'anterior, excepte la primera que la considerem tasca inicial. Per aquest fet a cada tasca només és considera la possibilitat de que hagi restriccions físiques i no lògiques. Les restriccions lògiques només les considerem a l'hora d'iniciar cada tasca, on hem de vigilar que no comenci una tasca sino finalitza la predecessora.



Dibuix 1. *Diagrama de Gantt*

El camí crític és la relació entre les franges negres que indiquen la durada total de cada tasca, incloent en el temps cadascuna de les seves subtasques.



Dibuix 2. *Diagrama de Pert*

En aquest cas el camí crític es pot comprovar que són els requadres negres que indica número de tasca.

7.

Hi ha quatre requisits o llibertats per a què un software pugui ser considerat de programari lliure:

- **Llibertat 0:** Poder executar el programa segons les necessitats de cadascú.
- **Llibertat 1:** Adaptar el programa (modificar-lo) segons les nostres necessitats tenint accés al codi Font.
- **Llibertat 2:** Redistribució de les còpies.
- **Llibertat 3:** Permetre la distribució del programa amb les millores que s'han realitzat, així tant usuaris com institucions es poden beneficiar del canvi i, fins tot, començar de nou el procés adaptant el programa i redistribuir-lo de nou amb els nous canvis.

La *General Public License (GPL)* i la *Lesser General Public License (LGPL)* tenen en comú que són software de llicència lliure. En el software de llicència lliure es pot classificar dos tipus: vírica i dèbil.

A la vírica podem trobar diversos tipus de llicències, entre elles la GPL; al contrari que la LGPL, que formaria part de la dèbil.

La principal diferència entre els dos, és que el víric (amb llicència Copyleft) garanteix que el software sempre estigui llicenciat de manera lliure, garantint les quatre llibertats bàsiques comentades anteriorment; en canvi el dèbil (Copyleft dèbil) permet la combinació de diversos tipus de llicències, encara que siguin llicències de tipus privatives. Tot i així, per a què pugui forma part d'un programa amb llicencia lliure, ha de garantir la llibertat de codi per tal de poder accedir al codi font.

8.

És una eina que s'utilitza per la extracció (**E**xtract), transformació (**T**ransformation) i càrrega (**L**oad) de dades. Aquesta eina es pot trobar a la **D**ata **W**arehoure (emmagatzematge de dades). S'utilitza per emmagatzemar dades en un mateix format i emmagatzemar-les relacionada amb un camp data (per conèixer l'antiguitat) al DW. Òbviament, aquesta eina carrega arxius temporals que no són necessaris i que conseqüentment es fan una eliminació posterior.

9.

Un CVS (Concurrent Versions System) és una aplicació client-servidor que manté el registre de tot el treball i els canvis en els arxius que formen un projecte i permet que diferents treballadors puguin modificar-lo. La seva llicència és GPL. S'utilitza per administrar versions i canvis sobre els arxius (normalment són arxius amb codi font, però es pot aplicar a qualsevol tipus d'arxiu). També té l'avantatge de recuperar l'arxiu a un estat anterior, gràcies al seu registre.

Les operatives d'ús:

- Lock (Bloquejar, Modificar, Desbloquejar): Dos treballadors que fan ús d'un mateix arxiu, primer un l'obre i el té bloquejat per l'altre, el modifica i després el desbloqueja per a que l'altre treballador faci el mateix procés.
- Merge (Copiar, Modificar, Mesclar): Es fa una còpia amb la qual modifica l'arxiu, després es mesclen per a unir els dos arxius.

Hi ha dos tipus d'arquitectures:

- Centralitzada: Utilitza el sistema client-servidor i en fa ús de còpies de referència i còpies de treball. Utilitzen operacions amb el servidor central amb la desavantatge que és necessari l'ús de xarxa.
- Distribuïda: Utilitza un sistema Peer-to-Peer (P2P) i només fa ús de còpies de treball. Utilitza operacions locals, per tant no és necessari l'ús de xarxa per operacions típiques o comunes.

10.

Tot el que s'ha fet es força interessant, no obstant, destacaria la part de la planificació del projecte, tant la part teòrica com la pràctica. Amb la part pràctica, amb la qual hem pogut planificar un projecte amb OPENPROJ i OPEN ERP, ha servit per conèixer el tipus de restriccions que hi ha (lògiques i físiques). M'ha semblat útil el tipus de diagrames que es podien generar (Gantt i Pert). En conclusió, tot això t'ensenya que una bona planificació del projecte es garantia d'èxit, encara que els projectes informàtics siguin dels més difícils per finalitzar-los degut al munt d'imprevistos que hi ha al llarg del temps.

I tant que servirà pel mercat laboral, i no només això, sino ja em serveix per la feina que es fa la universitat.

El que trobo a faltar de l'assignatura és treballar una mica més en la part de projecte pràctic. Segurament això sigui degut al poc temps que hi ha a l'assignatura.