TEMA 1
PRINCIPIS DE L'ENGINYERIA DEL SOFTWARE

1. Definició i objectius de l'ES.

- Definició de software.
- Evolució del software.
- Crisi del software: problemes i causes.
- Característiques del software.
- Definició d'ES.
- Objectius de l'ES.
- Aplicacions del software.

2. Procés, mètode i eina.

- Definicions.
- Activitats en el procés de desenvolupament de software.

3. Paradigmes del desenvolupament del software.

- Model lineal seqüencial (cicle de vida clàssic).
- Model de prototipat.
- Model evolutiu.
- Model en espiral.
- Model àgil.

Per començar... Què és el software?

VIQUIPÈDIA
L'enciclopèdia lliure

Programari

El **programari** (*software*, en anglès) és el conjunt dels programes informàtics, procediments i documentació que fan alguna tasca en un ordinador. Comprèn el conjunt sistemàtic dels programes d'explotació i dels programes informàtics que serveixen per a aplicacions determinades.[1] El terme inclou aplicacions com els processadors de text, programari de sistema com el sistema operatiu, que fa d'interfície entre el maquinari i les aplicacions, i finalment el programari intermediari, que controla i coordina sistemes distribuïts.

Moltes vegades, el terme és usat per contraposició a maquinari. S'usa el primer per a descriure la part lògica d'un sistema informàtica i el segon, per a descriure la part física.[2] A diferència del maquinari, el programari "no es pot tocar".[3]

Definició de Software

El software és:

1. **Instruccions** (programes) que quan s'executen proporcionen la funció i el rendiment desitjats.

2. **Estructures de dades** que permeten als programes manipular adequadament la informació.

3. **Documents** que descriuen l'operació i la utilització dels programes.

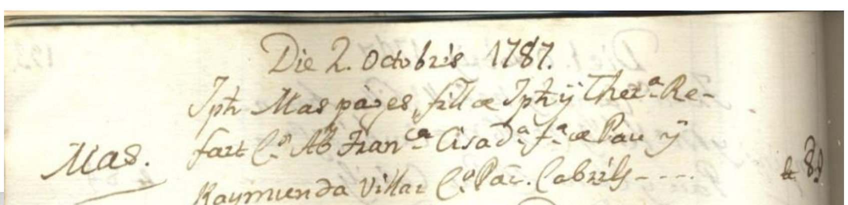
Sistema informàtic: compost per una sèrie d'elements que permeten dur a terme algún mètode, procediment o control a través del processament de la informació.

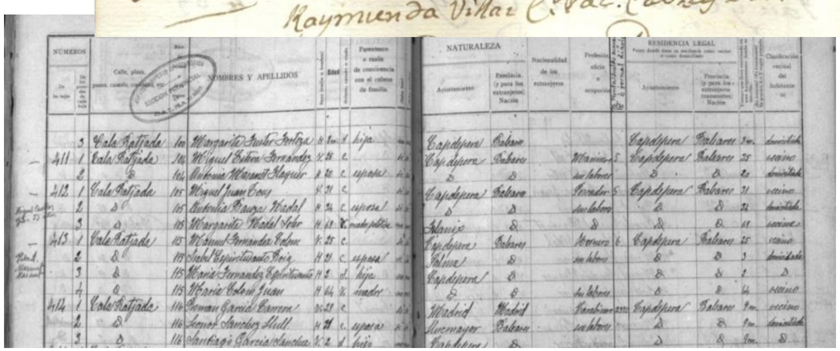
Components: hardware, software, persones, documentació, procediments i dades.

1.0

Un exemple de projecte: Facebook històric

Ens ha visitat el Dr. Xarbot, distingit arxiver, historiador i emprenedor. A partir de la digitalització i transcripció de registres de naixement, matrimonials, defunció, censos, informació cadastral, etc. emmagatzemada en arxius històrics i municipals, vol construir una xarxa social històrica amb persones, relacions, events, etc.


Die 2. Octobris 1787.
Mas.

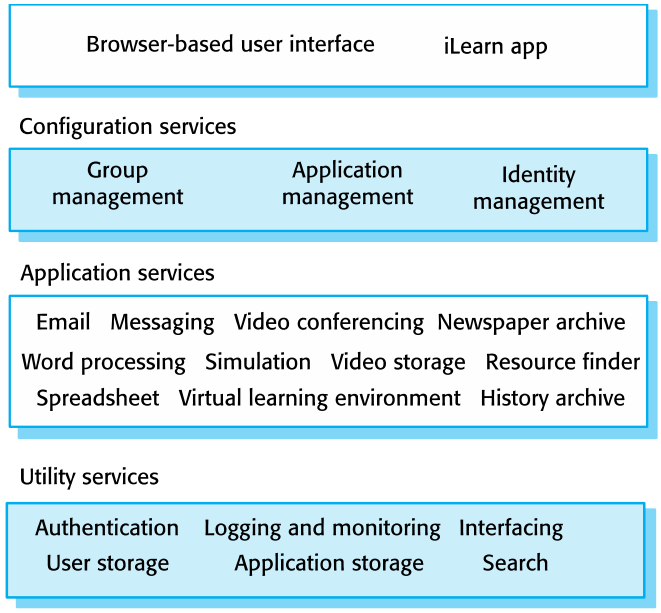


Què Fem?

- Sabem QUÈ hem de fer?
- Una vegada entès, COM ho farem?
- QUI ho farà? Com ens organitzarem?
- QUAN ho tindrem?
- QUANT costarà?
- Com assegurarem la qualitat?

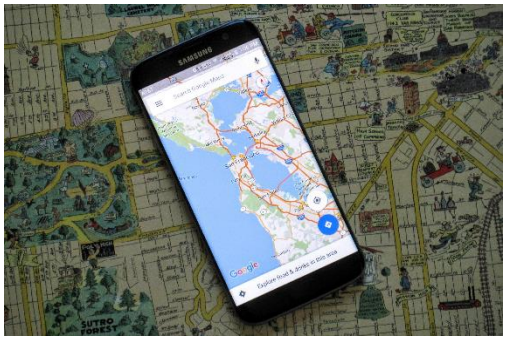
Exemple2: iLearn. Entorn digital d'aprenentatge

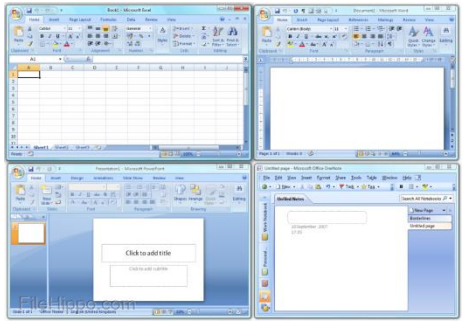
- Plataforma per entorn acadèmic que integri eines de propòsit general i dissenyades especialment per a l'aprenentatge, més un conjunt d'aplicacions adaptades a les necessitats dels estudiants usuaris del sistema.
- Els professors poden configurar les eines per les seves respectives aules virtuals. Les eines poden ser de propòsit general (fulls de càlcul, editors de text, email) o eines especifiques (ex. Lliurament de treballs, qüestionaris, …)
- Model Software as a Service (SaaS), on els serveis es poden incorporar de manera incremental, i es poden adaptar a les necessitats dels usuaris.



https://iansommerville.com/software-engineering-book/

Exemples de software quotidià que ha requerit un procés de desenvolupament

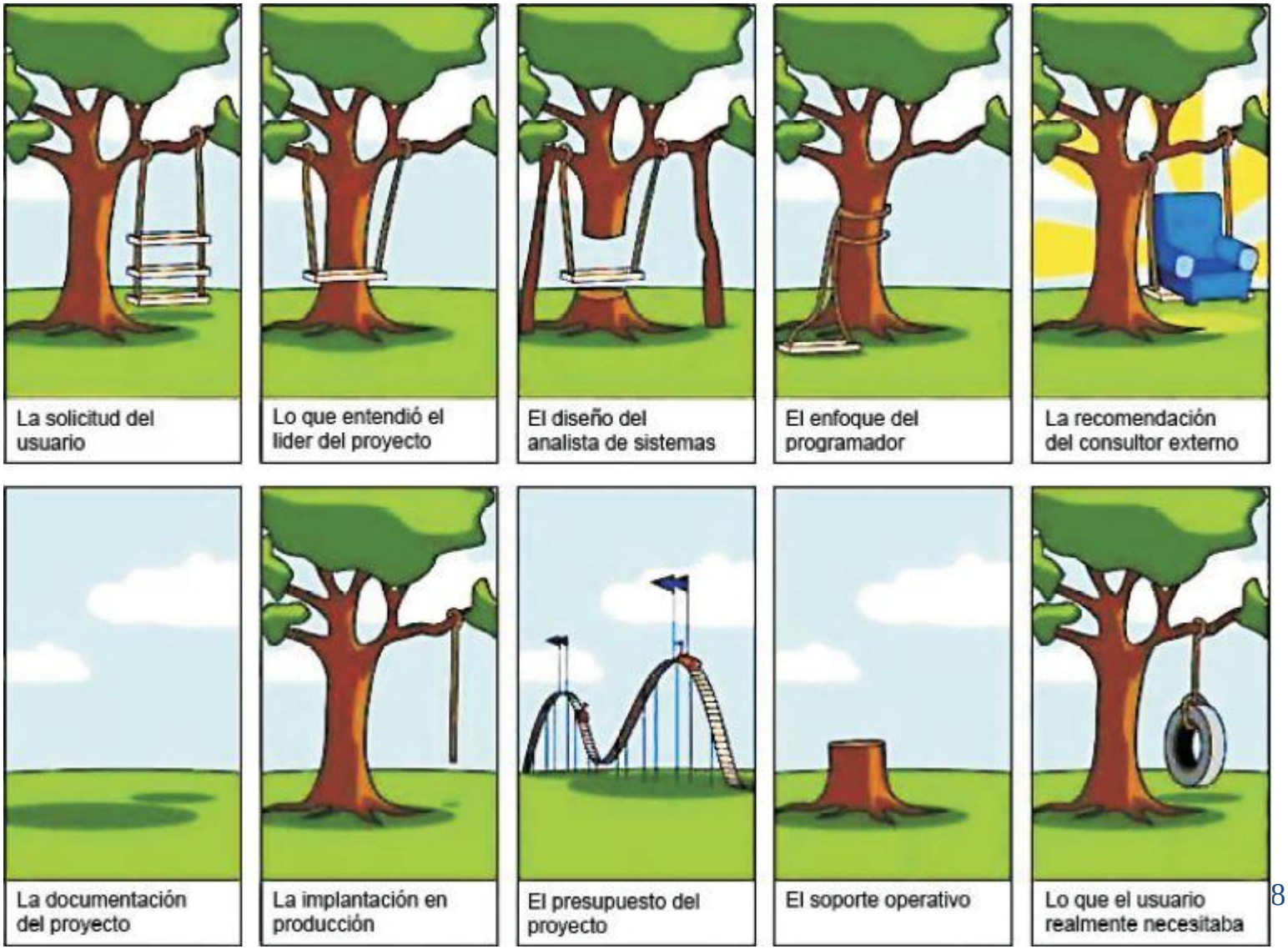




Les components de l'enginyeria del software

- QUI: els diferents rols de l'enginyer del software
- QUÈ: els resultats del treball (tasques)
- COM: metodologies, models de procés
- QUAN: fluxos de treball, planificació

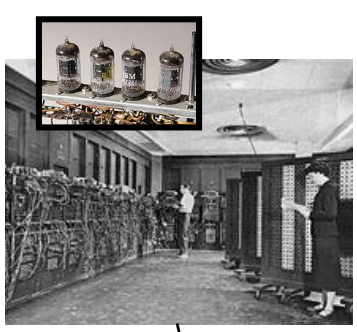
La realitat del procés de desenvolupament de software

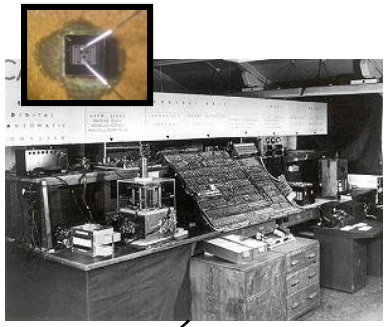


Productes de software

- **Productes genèrics**
  - Sistemes “stand-alone” que es publiciten i venen a **qualsevol client** que vol comprar-los.
  - Exemples: software de PC com programes d’edició, gràfics, fotografia, gestió de projectes, CAD, mercats específics (gestió matrícules per acadèmies d’ensenyament).
- **Productes a mida**
  - Software desenvolupat per encàrrec d’un **client específic** per satisfer les seves necessitats.
  - Exemples: sistemes de control empotrats (embedded) com ara control en producció industrial, software de control de tràfic, software de gestió d’un banc.

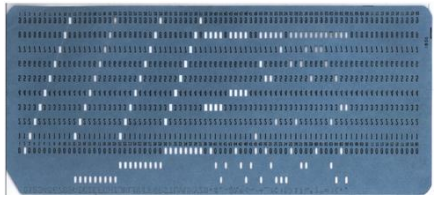
Evolució del Hardware i del Software





1940 1950 1960 1970 1980 1990 2000 2010

- SW es un afegit d'un únic HW.
- Sistemes poc interactius (batch).



- Millora interacció home màquina.
- Sistemes multiusuari.
- Primers Gestors de Base de dades.

- Ordinadors personals.
- HW comú per tothom. El SW marca la diferència.
- SW és un producte rendible !
- Processament distribuit.
- Xarxes

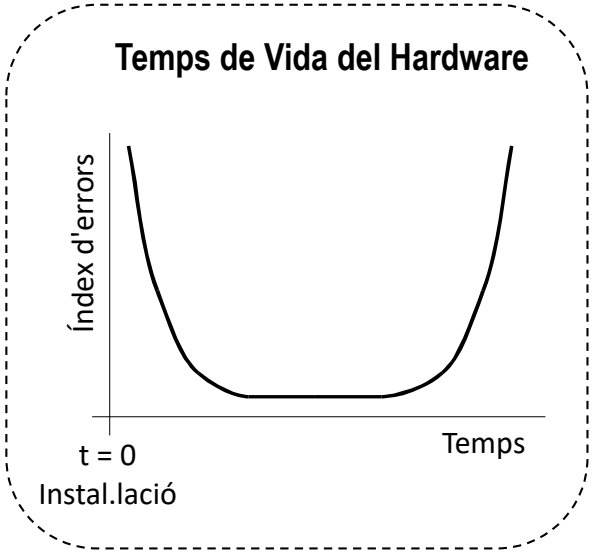
- Tècniques Orientades a Objecte.
- Computació paral.lela.
- Internet !

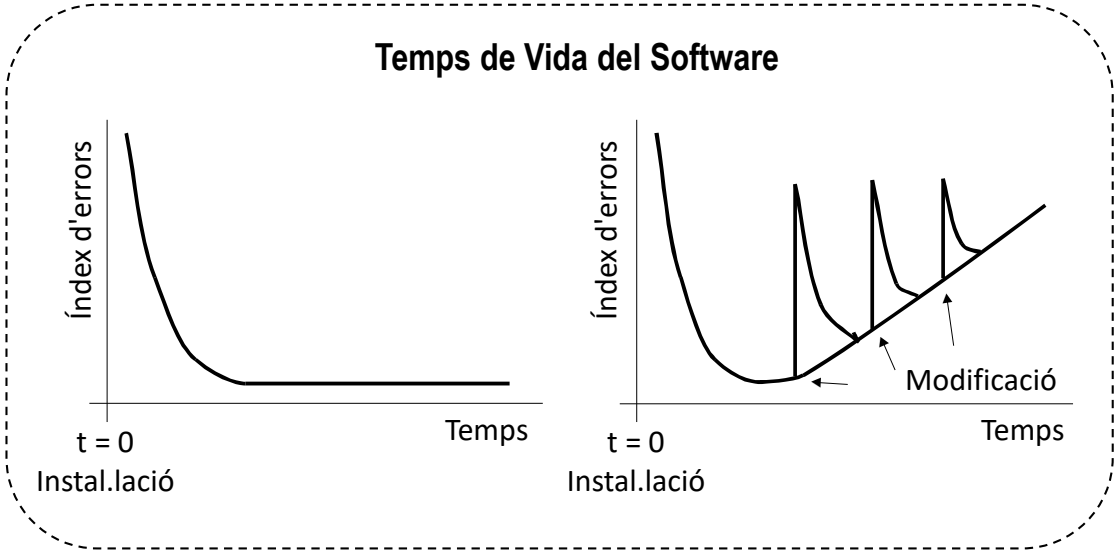
Costos de software

- Els costos del software sovint dominen els costos del sistema informàtic. Els costos de software en un PC solen ser més grans que el cost de hardware.
- El cost de manteniment de software és més alt que el de desenvolupament. Per a sistemes amb una llarga vida, els costos de manteniment poden ser diverses vegades els costos de desenvolupament.
- L'enginyeria del software es preocupa del desenvolupament de software rendible i eficient en costos.

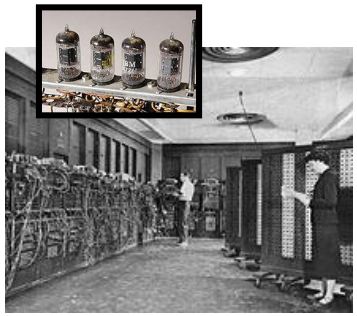
Característiques del Software vs Hardware

- Es desenvolupa, no es fabrica.
  - Costos del SW centrats a l'enginyeria (cost fabricació ≈ 0).
- Té un manteniment més complex.
  - No hi ha peces de recanvi.
- No es deteriora.
  - Però esdevé obsolet.





Efecte del Hardware en el Software



- **Velocitat de Procés: ↑ o ↓ ?**
- **Capacitat de memòria: ↑ o ↓ ?**
- **Mida en relació a potència de càlcul: ↑ o ↓ ?**
- **Cost en relació a la potència de càlcul: ↑ o ↓ ?**

Tenir ordinadors més potents ens ha tret feina de sobre ?

Pole Position (1982)

Need for Speed (1995)

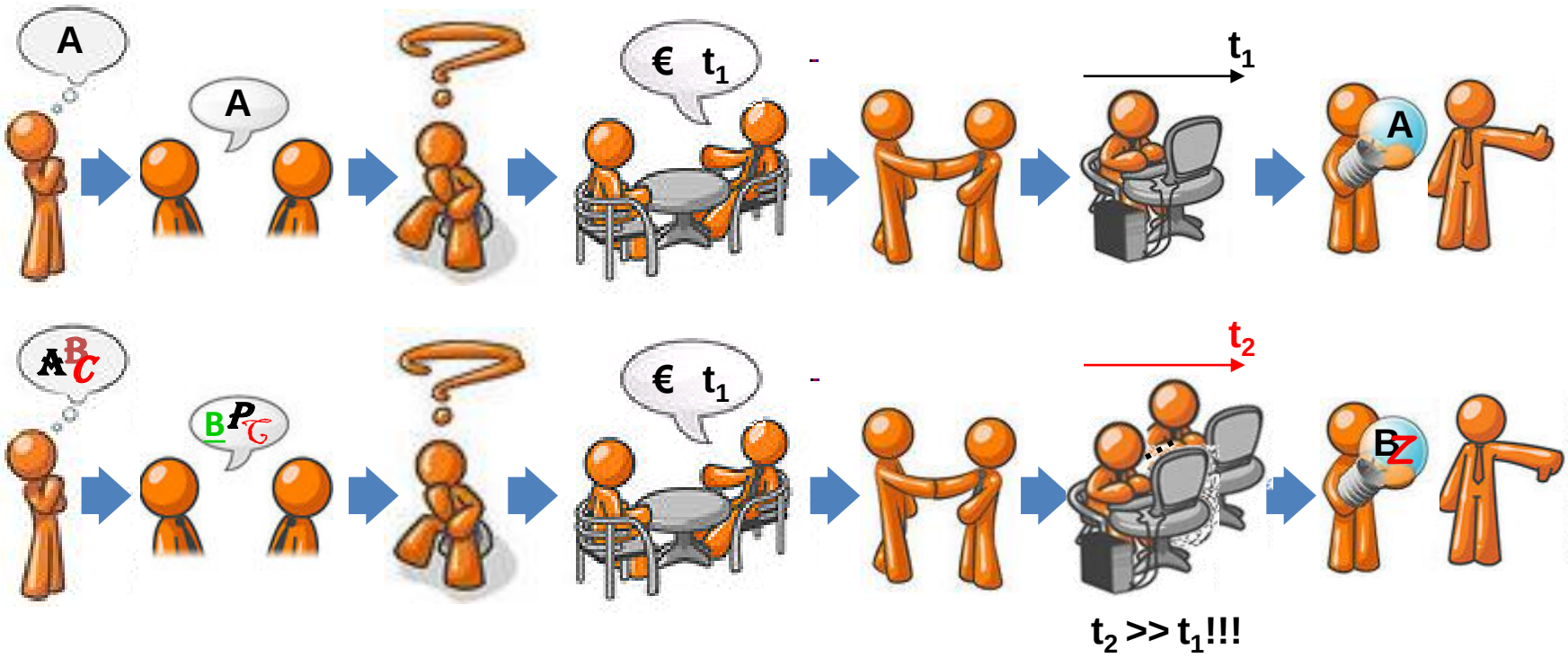
Need for Speed (2005)

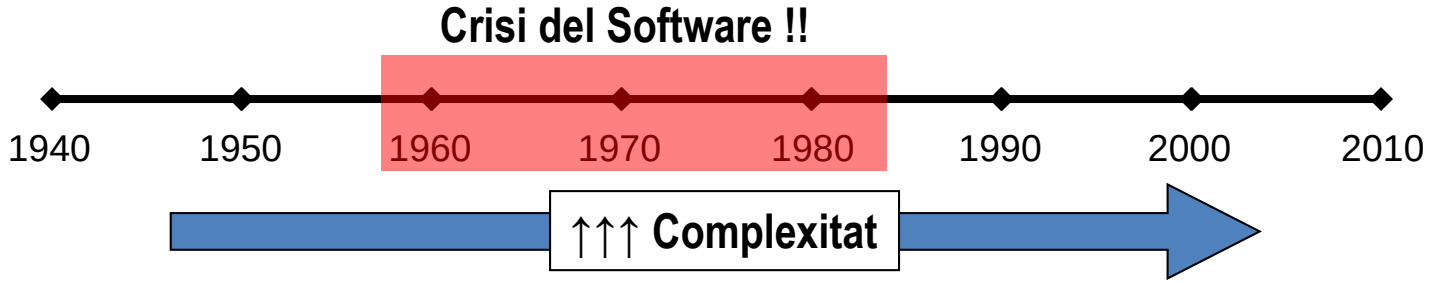
Need for Speed (2011)

Tenir ordinadors més potents ens ha multiplicat la feina !

↑ potència de càlcul → ↑ complexitat dels problemes plantejats.

Els projectes de software





La Crisi del Software

- Software…
  - Fora de pressupost.
  - Amb dates de lliurament incomplertes (si s'acaben entregant…).
  - Aportant una solució ineficient i/o amb errors.
  - Insatisfent requisits.
  - Molt complexe de mantenir (gestió del canvi molt complexa).
- Causes (entre altres…)
  - Falta de metodologia, mals hàbits.
  - Aplicació de tècniques clàssiques sense èxit.
  - Comunicació no efectiva entre el client i el desenvolupador.
  - Prova del software insuficient.
  - Detecció d'errors molt tardana.
- Solució ? ➡ Enginyeria del Software (60's)

L'enginyeria del Software

Definició:

- L'aplicació d'un enfocament sistemàtic, disciplinat i quantificable per al desenvolupament, operació i manteniment del software. És a dir, l'aplicació d'un procés d'enginyeria al software.

1. L'estudi d'enfocament com els indicats a (1)

IEEE: Institute of Electrical & Electronics Engineers

Objectiu: Desenvolupar amb <u>èxit</u> SW de <u>qualitat</u> acomplint objectius de cost !

- satisfà l'usuari
- funciona impecablement
- és fàcil d'usar
- és fàcil de mantenir

Objectius de l'enginyeria del software

- **Cost:** senzill de calcular.
- **Qualitat:** depèn de factors com ara:

Mètrica de Qualitat del Software

Factors operatius (utilització)
- **Correctesa:** fins a quin punt el programa satisfà les especificacions inicials.
- **Exactitud:** és la propietat que defineix amb quin grau compleix el software els requisits establerts.
- **Eficiència:** és un factor que es refereix en tots els sentits a l'execució del software, i inclou factors com ara temps de resposta, requisits de memòria i capacitat de processament.
- **Integritat:** Control d'accessibilitat al software per persones no autoritzades.

Factors de revisió (canvis)
- **Usabilitat:** és l'esforç necessari per aprendre i utilitzar el software de manera correcta.
- **Mantenibilitat:** és l'esforç necessari per detectar i corregir errors en un programa que estigui instal·lat.
- **Testabilitat:** és l'esforç necessari per provar, per assegurar que el sistema o un mòdul realitzi el seu propòsit.
- **Flexibilitat:** és l'esforç necessari per modificar un programa un cop instal·lat.

Factors de transició (adapt. a nous entorns)
- **Transportabilitat:** és l'esforç necessari per traspassar el software des d'una configuració hardware a una altra.
- **Reusabilitat:** fins a quin punt parts del software poden ser reaprofitades en altres aplicacions.
- **Interoperativitat:** és l'esforç necessari per acoblar el sistema amb altres sistemes.

Categories del software

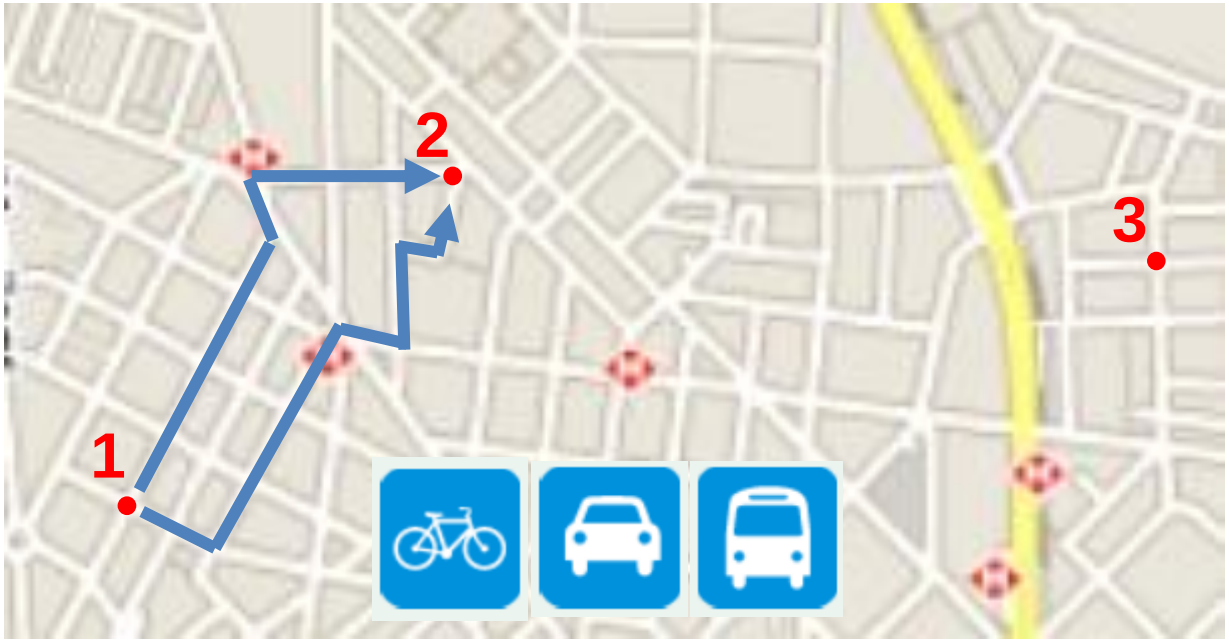
- **Software de sistemes.** Programes escrits per servir a altres programes (compiladors, editors, gestors de comunicacions, etc.). Forta interacció amb el hardware.
- **Aplicacions software.** Programes "stand-alone" per necessitats específiques.
- **Software de temps real.** Software que mesura/analitza/controla events del món real a mesura que esdevenen. Components: adquisició de dades, anàlisi, control/sortida, monitorització.
- **Software de gestió.** Processament d'informació comercial.
- **Software d'enginyeria i científic.** Software de càlcul numèric, biologia molecular, física de partícules, simulació, etc.
- **Software empotrat.** Software de només lectura que controla productes i sistemes dels mercats industrials i de consum (ordinador d'un cotxe, control d'un forn, etc.)
- **Software d'intel·ligència artificial.** Algorismes no numèrics per resoldre problemes complexos per als quals no és adequat el càlcul o l'anàlisi directa. Visió, robòtica, jocs, reconeixement de parla, etc.
- **WebApps.** Aplicacions "en xarxa" basades en entorns web.
- **Software de línia de producte.** Mercat particular en què s'adreça un volum massiu de consumidors (procesadors de text, grafics, gestors bases de dades).

Noves categories de software

- **Computació distribuïda/ubiqua.** Computació pervasiva, ubiqua, distribuïda deguda a xarxes wireless. Com permetre als dispositius mòbils, ordinadors personals, sistemes industrials comunicar-se a través de xarxes.
- **Cloud computing.** SaS (software as a service). La web com una màquina de càlcul.
- **Big data.** Processament de dades massives i detecció de patrons en elles.
- **Open source.** Software de codi obert “lliure” per a la comunitat.
- Altres:
  - Data mining
  - Grid computing
  - Cognitive machines
  - Software per nanotecnologies

Processos de Desenvolupament de Software

- Hi ha múltiples propostes de com s'ha de desenvolupar SW.
- La base està en definir un procés a seguir



Procés: Descriu quines activitats realitzar per obtenir SW de qualitat.
Mètode: Indica com s'han de dur a terme les activitat del procés.
Eina: Proporciona suport automàtic o semiautomàtic a procés i mètodes.

CASE: Computer Aided Software Engineering

Procés, mètode i eina

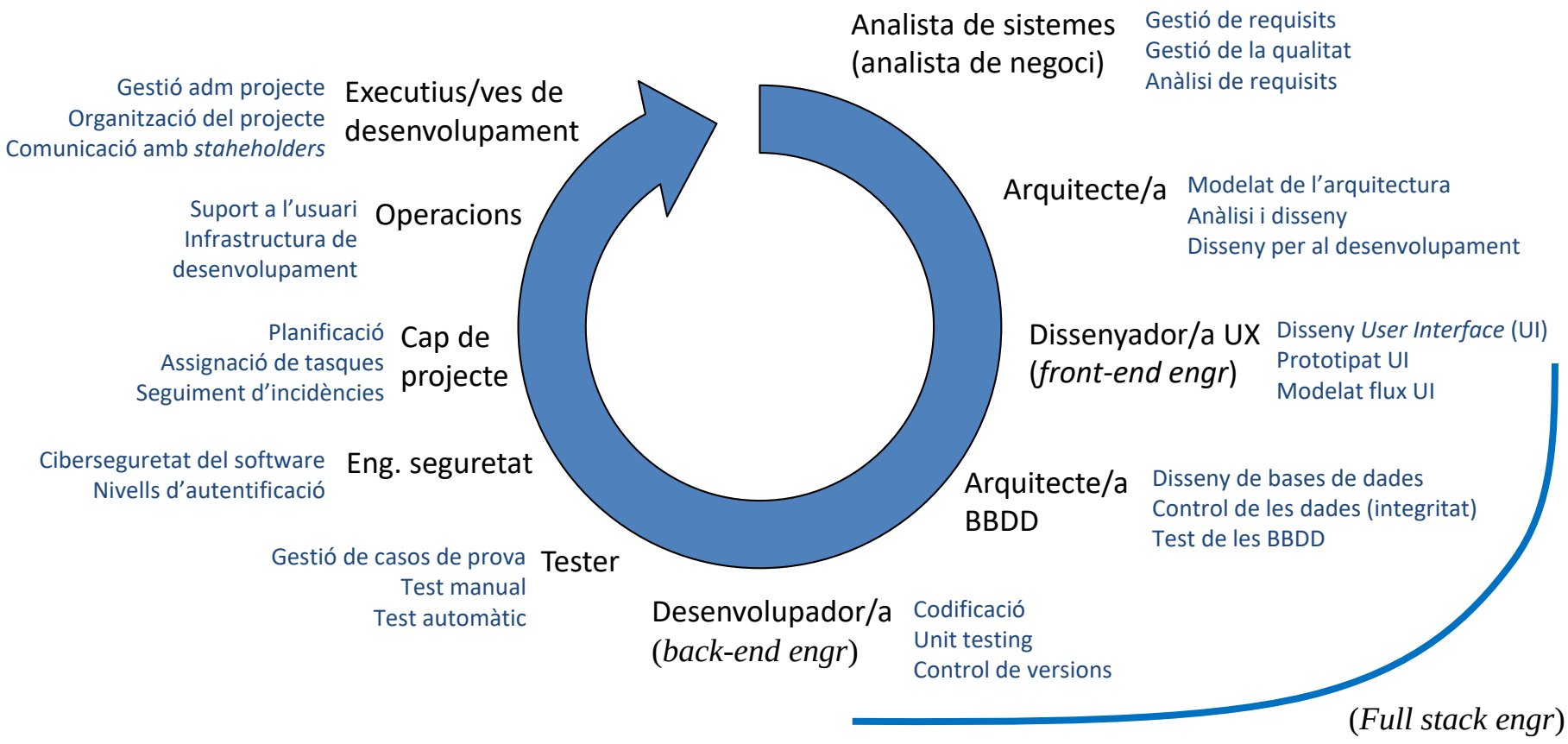
- **Procés:** defineix un marc de treball per a un conjunt d'àrees que s'han d'establir per a l'entrega efectiva de la tecnologia de l'ES. És el context en què s'apliquen els mètodes tècnics, es produeixen els resultats del treball (models, documents, dades, informes, formularis, etc.), s'estableixen fites, s'assegura la qualitat i el canvi es gestiona adequadament.
- **Mètode:** indica com construir tècnicament el software. Ofereixen tècniques de modelatge per a les diferents etapes del procés.
- **Eina:** proporcionen suport automàtic o semiautomàtic per al procés i per als mètodes. => CASE.

Fases (activitats) en el procés de desenvolupament de software

- Anàlisi de requisits.
  - Comprensió del problema.
  - Especificació de requisits.
- Disseny.
  - Disseny preliminar (detecció de components o mòduls).
  - Disseny detallat (lògica interna de cada mòdul).
- Codificació.
- Prova.
  - Proves d'unitat.
  - Proves d'integració.
  - Proves de sistema (el sistema satisfà els requisits).
  - Proves d'acceptació

  => Caixa blanca i caixa negra.

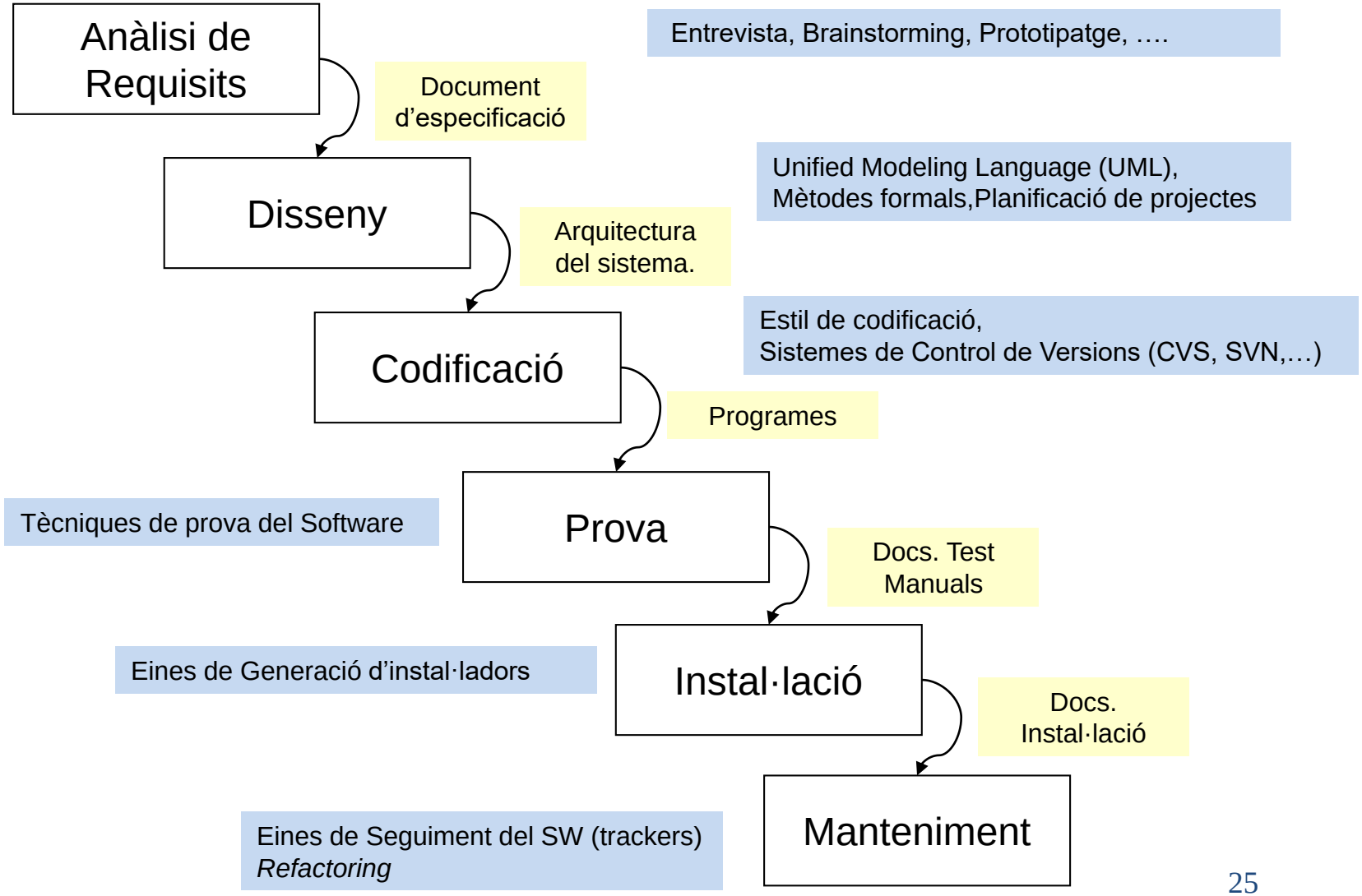
L’enginyer del software: rols i responsabilitats



Paradigmes del procés de desenvolupament de software

- Model lineal seqüencial (cicle de vida clàssic o salt d'aigua).
  Enginyeria i anàlisi de sistemes (anàlisi de viabilitat), anàlisi de requisits, disseny, codificació, prova i manteniment.
- El model de construcció de prototipus.
- Model evolutiu.
  Té en compte el caire evolutiu del software (és un model iteratiu que permet desenvolupar versions cada vegada més completes).
  - Model incremental.
  - Model en espiral.
  - Models Àgils (Scrum, Lean, Kanban, DevOps …)
- Altres.
  - Model de desenvolupament ràpid d'aplicacions (DRA).
    Construcció basada en components.
    És un model lineal seqüencial però amb un temps extremadament curt. Utilitza eines CASE.
  - Model d'assemblatge de components.
    Partint d'una llibreria de classes inicial, es combinen les que són útils i se n'implementen de noves que s'afegeixen a la llibreria.
  - Model de desenvolupament concurrent.
    Modelitza les etapes del procés d'ES com a conjunt d'estats i transicions entre ells.
  - Model de mètodes formals.
    Permet una especificació matemàtica del software.

Paradigma Lineal-Sequencial (cicle de vida clàssic)



Cicle de vida clàssic

Avantatges:

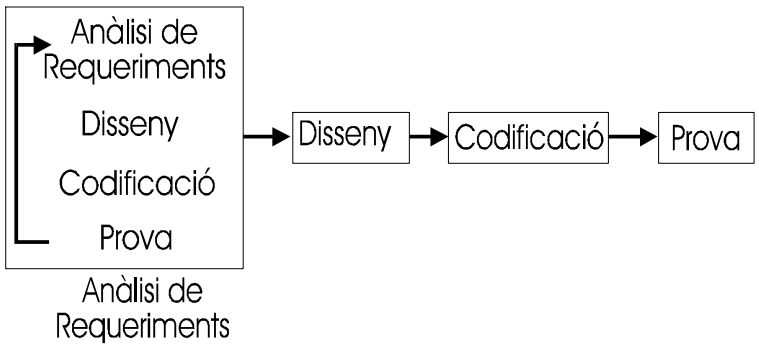
- Inici-Final de cada etapa definit → Progrés del projecte mesurable.
- Promou una documentació detallada.
- Acord signat dels requisits del sistema.

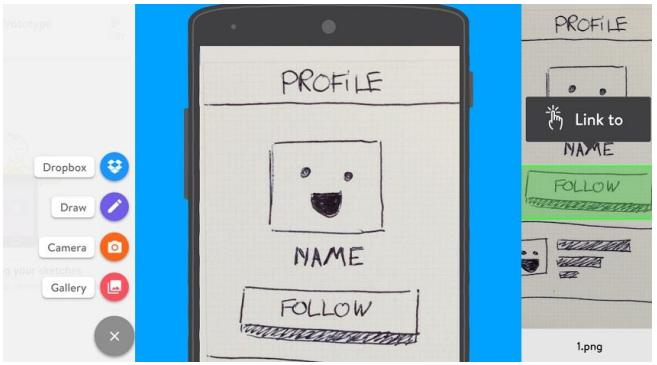
Inconvenients:

- Un projecte real mai no és tan marcadament seqüencial.
  Cal contemplar la possibilitat de tornar a estadis anteriors de desenvolupament.
- Dificultat d'establir explícitament tots els requisits al principi del projecte.
  El client pot no tenir-los clars o poden canviar en el decurs del procés.
- El client s'ha d'esperar a veure el resultat al final.
  Tota la planificació s'orienta a un únic dia de lliurament.
- No permet un desenvolupament per etapes.
  A vegades seria millor fer una part del sistema i després, millorar-lo i completar-lo, però això no es pot fer si tots els requisits s'han de conèixer a priori.

Construcció de prototipus

- Per millorar els tres primers inconvenients del cicle de vida clàssic.
- Es desenvolupa a partir d'un conjunt de requisits coneguts (objectius generals): recollida d'objectius globals, disseny ràpid, construcció del prototipus, test del prototipus.
- Formes:
  - En amplada: totes les funcions estan limitades o amb algorismes ineficients.
  - En profunditat: un subconjunt del producte final, funcions problemàtiques.
- **Inconvenients:**
  - El client ho veu com software definitiu.
  - Prototipus fet amb recursos inadequats. El desenvolupador no ha de "deixar-se dur" per la comoditat.





Model evolutiu

- Intenta solucionar les quatre limitacions del model del salt d'aigua, intentant combinar els beneficis del prototipatge i del cicle de vida clàssic.
- El software es fa per increments i cada increment afegeix una nova funcionalitat al sistema, a més de fer les extensions del disseny necessàries. Cada increment es prova (això suposa un avantatge ja que el sistema rep una prova més exhaustiva).
- ***Llista de control del projecte*** (tasques ordenades que ha d'incorporar la implementació final). A cada pas es treu una tasca de la llista, es dissenya i s'implementa la solució i s'analitza el sistema obtingut a fi d'actualitzar la llista.
  => *Fase de disseny, fase d'implementació i fase d'anàlisi.*
- **Limitacions:**
  - El manteniment deixa de ser una etapa com a tal i es fa al llarg de tot el procés. Com es quantifiquen les modificacions demanades pel client?
  - Dificultat en marcar la fi de la inclusió de requisits a la llista de control per part del client.

Cap als models Àgils de Desenvolupament de Software

- Hi ha múltiples propostes
  - Com es planifiquen les iteracions ?. Què es programa primer ?
  - Com es treballa en equip ?
  - Com es documenta el desenvolupament ?

Burocràtiques, o tradicionals → *Àgils*

Capability Maturity Model Integration (CMMI)

Rational Unified Process(RUP)

eXtreme Programming (XP)

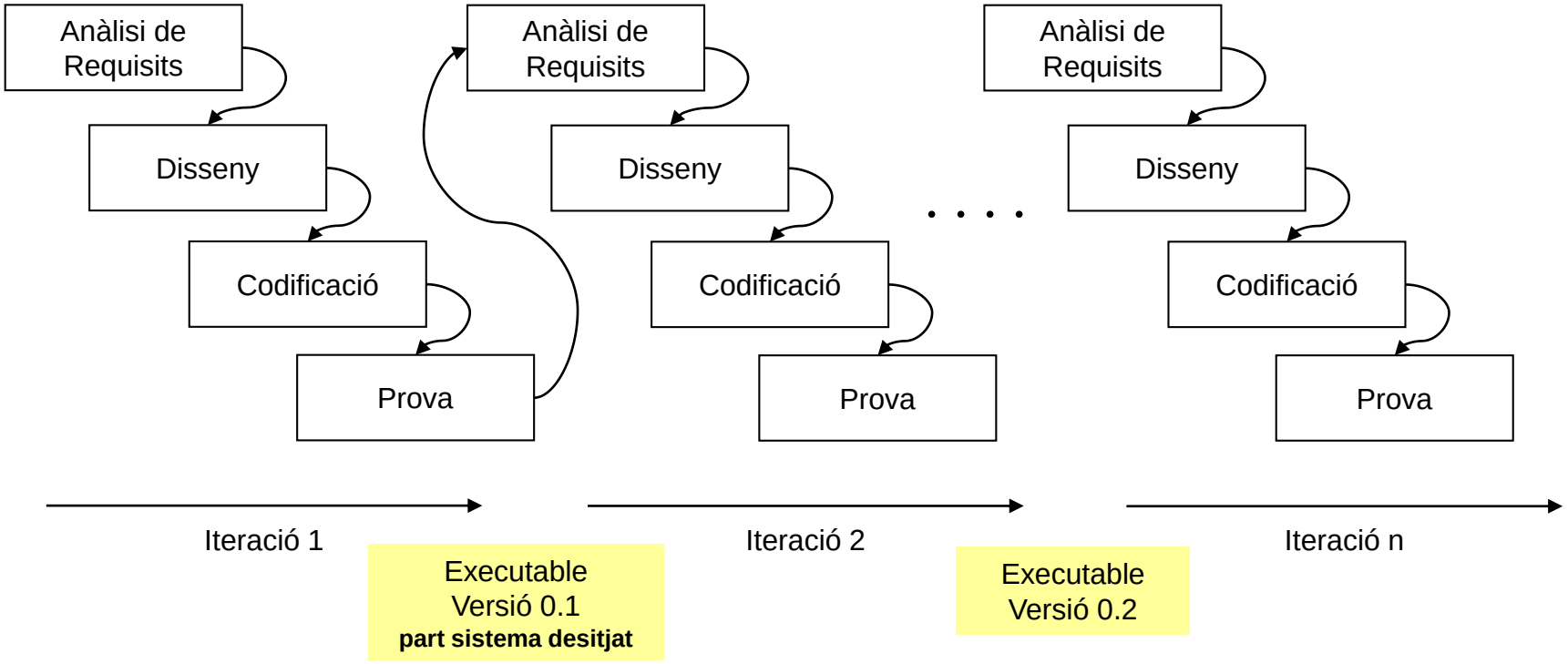
Scrum

Crystal

http://agilemanifesto.org/

Paradigma Evolutiu: visió incremental

El software es fa incrementalment, i cada increment afegeix noves funcionalitats al sistema (és a dir, satisfà requisits), a més de fer les extensions de disseny necessaries.

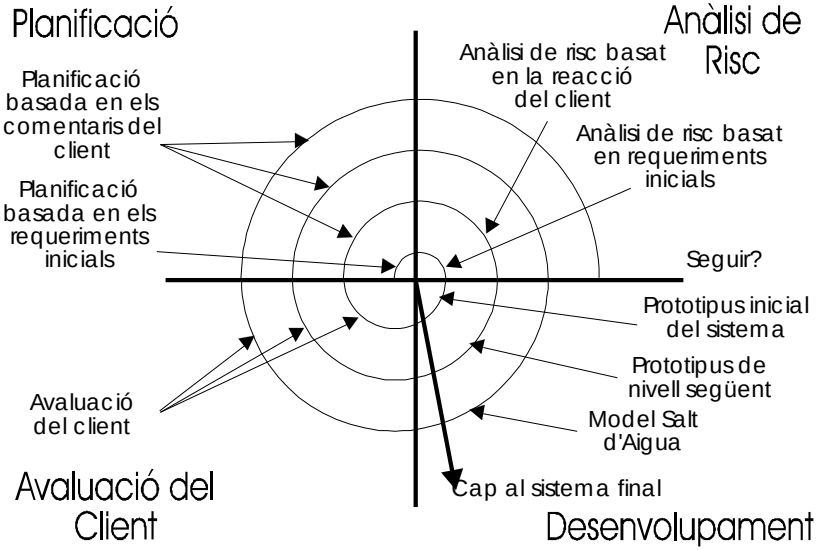


Cada increment es prova → detecció de problemes més precoç.

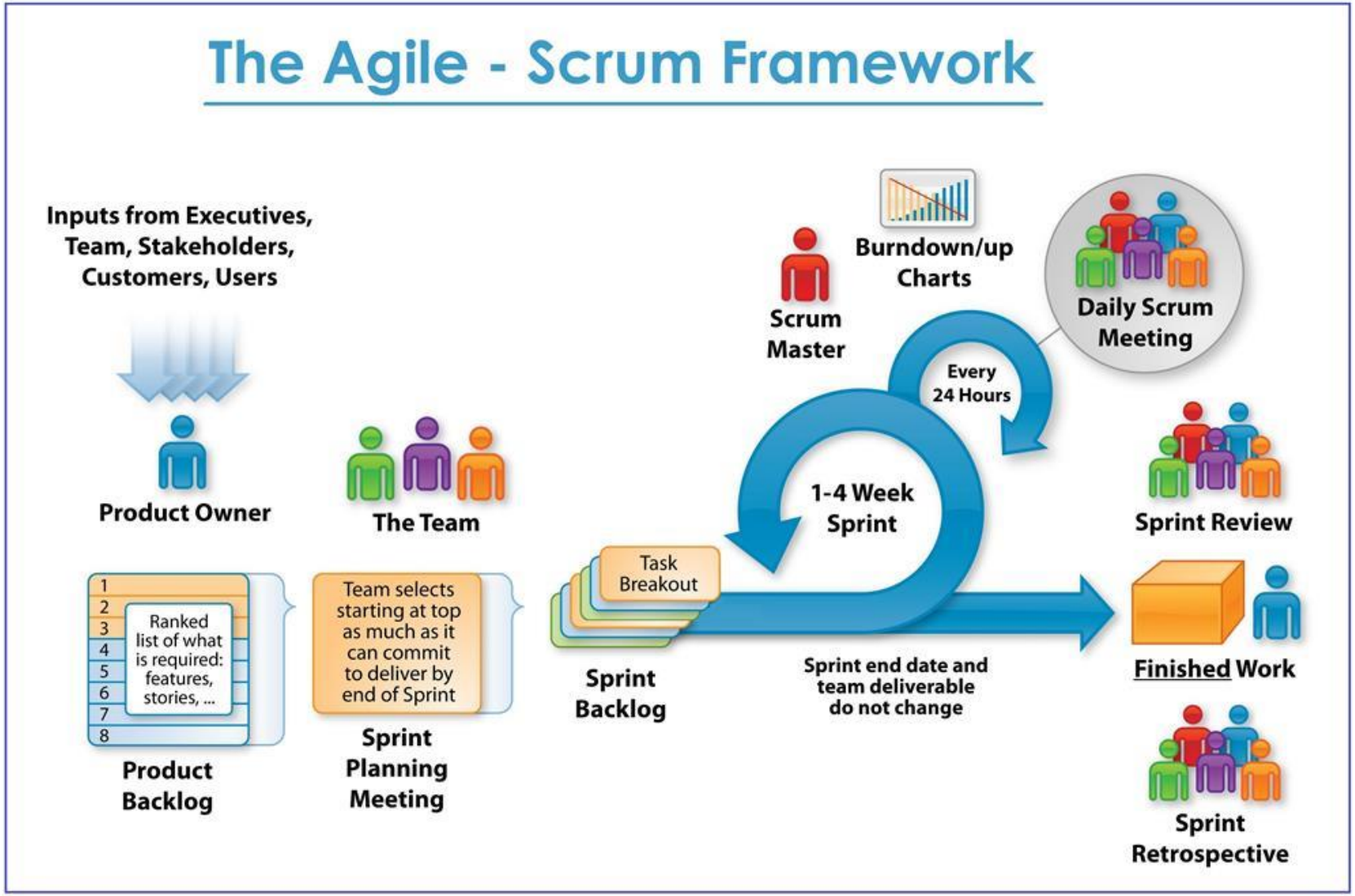
Molt feedback amb el client (amb els inconvenients que això pot implicar…)

Paradigma Evolutiu: visió en espiral

- Una altra manera de veure'l és com un **model en espiral**:
  - Afegeix una etapa d'avaluació del risc (circumstàncies potencialment adverses que poden perjudicar el procés de desenvolupament i la qualitat del producte).
  - Es desenvolupen cíclicament sis etapes: comunicació amb el client, planificació, anàlisi del risc, enginyeria (disseny), construcció, i avaluació del client.
  - A cada cicle s'aconsegueix una versió més sofisticada del producte.
  - L'espiral pot mantenir-se operativa sempre.

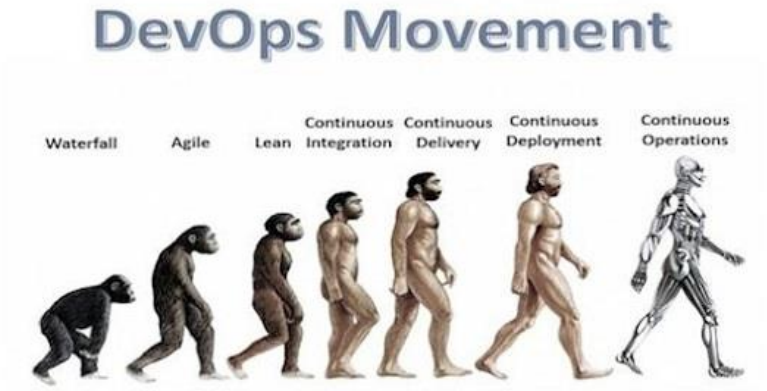


SCRUM


The Agile - Scrum Framework
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting

DevOps

- **DevOps** és una metodologia basada en increments curts i que aplica integració i desplegament continuat. És acrònim de development (Dev) i Operations (Ops), és a dir, apropa les activitats de desenvolupament i d'operacions (infrastructura, monitoreig de rendiment, …).
- **Integració contínua:** les compilacions i les proves d'unitat automatitzades asseguren que els errors es detecten de forma precoç, alhora que permeten als provadors tenir accés immediat a la versió de codi més actual. Entorns com *Jenkins* permeten aquestes pràctiques.
- **Lliurament continu:** La integració contínua s'estén amb proves automatitzades dins d'un entorn objectiu, tenint una versió executable del codi integrat que es passa a entorn de prova i pot pasar a producció de manera fiable.
- **Desplegament continu:** les proves i lliurament automatitzats permeten que les funcionalitats verificables es puguin passar immediatament a la producció. Si algun usuari té algun problema, es torna enrera perquè el desenvolupador ho corregeixi.
- **Operacions contínues:** redueix o elimina la necessitat de temps morts planificats, com ara el manteniment programat. S'alimenta dels predecessors (integració, lliurament, desplegament continus). Exemples són els web services o els dockers que permeten que les aplicacions finals tinguin automàticament les actualitzacions, que es van afegint en cicles curts.



FAQ d'enginyeria del software

Pregunta	Resposta
Què és software?	Programes d'ordinador, estructures de dades i documentació associada. Els productes de software poden ser desenvolupats per a un client particular o per a consum massiu.
Quins són els atributs d'un bon software?	El bon software ha de donar el rendiment i la funcionalitat requerida per l'usuari, i hauria de ser mantenible i usable.
Què és enginyeria del software?	L'enginyeria del software és una disciplina de l'enginyeria que es preocupa de tots els aspectes de la producció de software.
Quines són les activitats fonamentals de l'enginyeria del software?	Anàlisi de requisits, disseny, codificació, test.
Quina és la diferència entre enginyeria del software i ciències de la computació?	Les ciències de la computació se centren en la teoria i fonaments; l'enginyeria del software s'ocupa dels aspectes pràctics de desenvolupament i distribució de programari útil.
Quina és la diferència entre enginyeria del software i enginyeria de sistemes?	L'enginyeria de sistemes s'ocupa de tots els aspectes del desenvolupament de sistemes basats en ordinador incloent hardware, software i enginyeria de processos. L'enginyeria del software és una part d'aquest procés més general.

FAQ d'enginyeria del software

Pregunta	Resposta
Quins són els reptes principals de l'enginyeria del software?	Fer front a la diversitat creixent, exigir temps de lliurament reduïts i desenvolupar programari de confiança.
Quins són els costos de l'enginyeria del software?	Aproximadament el 60% dels costos de programari són costos de desenvolupament, el 40% són costos de prova. Per al programari personalitzat, els costos d'evolució solen superar els costos de desenvolupament.
Quines són les millors tècniques i mètodes d'enginyeria del software?	Si bé tots els projectes de programari s'han de gestionar i desenvolupar professionalment, diferents tècniques són adequades per a diferents tipus de sistemes. Per exemple, els jocs sempre s'han de desenvolupar mitjançant una sèrie de prototips, mentre que els sistemes de control crítics de seguretat requereixen una especificació completa i analitzable. Per tant, no podem dir que un mètode sigui millor que un altre.
Què ha significant el www i les plataformes mòbils per a l'enginyeria del software?	El www i mòbils ha suposat la disponibilitat de serveis de programari i la possibilitat de desenvolupar sistemes basats en serveis altament distribuïts. El desenvolupament de sistemes basats en web ha suposat importants avenços en llenguatges de programació i reutilització de programari.