Taller Cooperativisme de Plataforma

Estrategies de Desenvolupament

Autoria: Enrico Stano (CoopDevs),
Pau Pérez (CoopDevs), Fèlix Casanellas (Colectic)

Barcelona Activa

CONTINGUTS

Repassem els components de l'ecosistema en què es basa una plataforma tecnològica, analitzem tant els elements tècnics com els socials. Veiem els fonaments de desenvolupament en obert de programari lliure amb equips distribuïts i comunitats obertes, arquitectures de xarxes distribuïdes i el seu potencial.

CONCEPTES CLAU

- Plataformes tecnològiques
- Arquitectura d'apps distribuïdes
- Bases de dades
- Apps Mòbils
- Metodologies de desenvolupament
- Operacions i administració de sistemes
- Repercussió procomú

ECOSISTEMA DE PLATAFORMA - GLOSSARI

Client d'escriptori: Usuari/ària que accedeix a la plataforma a través d'un ordinador (de sobretaula o portàtil).

Sistema operatiu: Programari que estableix la comunicació entre les funcionalitats del maquinari del dispositiu i les necessitats de les aplicacions instal·lades
Exemples: En dispositius de sobretaula GNU/Linux, Microsoft Windows, Mac OS ... en dispositius mòbils Android, iOS ...

Aplicació/programa: Element de programari desenvolupat per executar una tasca concreta emprant els components del dispositiu on s'instal·la.
Exemples: Processador de textos, editor de vídeo, aplicació de missatgeria, videojocs, antivirus...

Client mòbil: Usuari/ària que accedeix a la plataforma a través d'un dispositiu mòbil (telèfon intel·ligent o tauleta).

Navegador web: Programa dedicat sol·licitar i interpretar continguts a un servidor web i mostrar-los a l'usuari/ària. Interpreta els llenguatges HTML, CSS i Javascript.
Exemples: Firefox, Chrome, Safari i Internet Explorer.

Plataforma web: Programari pensat perquè els usuaris/àries hi accedeixin a través d'un navegador web. La plataforma utilitza serveis i programari del servidor on s'allotja i si convé també d'altres.
Exemples: Botigues virtuals, portals informatius, webmail, editors on-line, cercadors...

Backoffice: Espai de la plataforma web que permet a les administradores d'aquesta incorporar-hi continguts, modificar-ne el comportament, activar i desactivar seccions i funcionalitats...

- **API (Interfície de Programació d'Aplicacions):** Interfície que permet que diferents elements de programari es puguin enviar informació mútuament.
  Exemple: Una xarxa social implementa una API per tal que altres aplicacions puguin publicar-hi a través dels perfils dels seus usuaris.
  - A diferència de la web clàssica, no es tracta d'una interfície d'usuari que permet la interacció persona-màquina, sinó que només proporciona les dades per a que les processi el client, el qual, després generarà la interfície d'usuari adequada per permetre aquesta interacció. Exemple https://jsonplaceholder.typicode.com/posts
- **Temes/Aparences:** Components de programari que permeten a les plataformes construïdes modularment modificar la seva aparença gràfica aplicant un disseny determinat.
- **Mòduls/Plugins:** Components de programari que permeten a les plataformes construïdes modularment incorporar noves funcionalitats.
  Exemple: Formulari de contacte, galeria d'imatges, cercador, agenda d'esdeveniments, passarel·la de pagament...
- **Gestor de continguts:** Programari de propòsit general ampliable modularment que porta incorporades les funcionalitats bàsiques per desplegar un gran nombre de projectes diferents. Funcionalitats com ara la gestió de reols i usuaris, tipus de continguts, seguretat ...
  Exemple: Wordpress, Drupal, Moodle, Prestashop...
- **Llenguatge de programació:** Conjunt de regles semàntiques i sintàctiques que permeten descriure estructures de dades i comportaments lògics que l'ordinador acaba interpretant.
  Exemple: PHP, Java, Python, Ruby, Javascript, C++...
- **Llenguatges de marcat:** Conjunt de regles semàntiques i sintàctiques permeten definir estructures de dades de forma jerarquitzada.
  Exemple: HTML, CSS, XML, JSON...
- **Servidor:** Equipament informàtic accessible a través de la xarxa amb una alta disponibilitat (24/7) que ofereix un seguit de serveis en funció del programari que tingui instal·lat i executant-se.
- **Servei web:** Programari que s'executa en un servidor dedicat a servir plataformes web. El servei rep les peticions dels usuaris/àries interpreta el codi de la plataforma i ofereix els continguts en format web.
- **Servei de correu:** Programari que s'executa en un servidor i s'encarrega de gestionar les bústies de correu dels seus usuaris/àries i a realitzar els enviaments.
- **Servidor de proves:** Servidor emprat per a testejar i integrar desenvolupaments abans de posar-los en el servidor de producció a l'abast del públic general.

- **Servei de DNS:** Programari dedicat a resoldre noms de domini retornant l'adreça IP del servidor que té assignat el domini sol·licitat.
  Exemple: El servei rep una petició per resoldre el domini test.com i el retorna que la IP assignada és la 69.172.200.235

- **Servei de base de dades:** Programari encarregat de gestionar estructures de dades normalment organitzades en taules que poden mantenir relacions entre elles. El servei permet diferents operacions sobre les dades tals com insercions, modificacions, consultes...
  Exemple: MySQL, PostgreSQL, MongoDB...

- **Desenvolupament a mida:** Quan un projecte requereix unes estructures de dades o uns patrons de comportament molt específics que no es poden satisfer a través de solucions estandarditzades requereixen un desenvolupament a mida. Es pot desenvolupar a mida des d'un component determinat fins a una solució sencera.

- **Compilador:** Programari que transforma el codi escrit en un llenguatge de programació *compilable* a un llenguatge maquin només interpretable per les màquines.

- **Binari executable:** Fitxer executable interpretable només per una màquina que conté un programa o aplicació.

- **Codi interpretable:** Fitxer de codi que no requereix un compilador per executar-se, es pot interpretar tal com s'ha escrit.

- **Codi compilable:** Fitxer de codi que requereix ser compilat per tar de poder-se executar.

ECOSISTEMA DE PLATAFORMA - AGENTS IMPLICATS

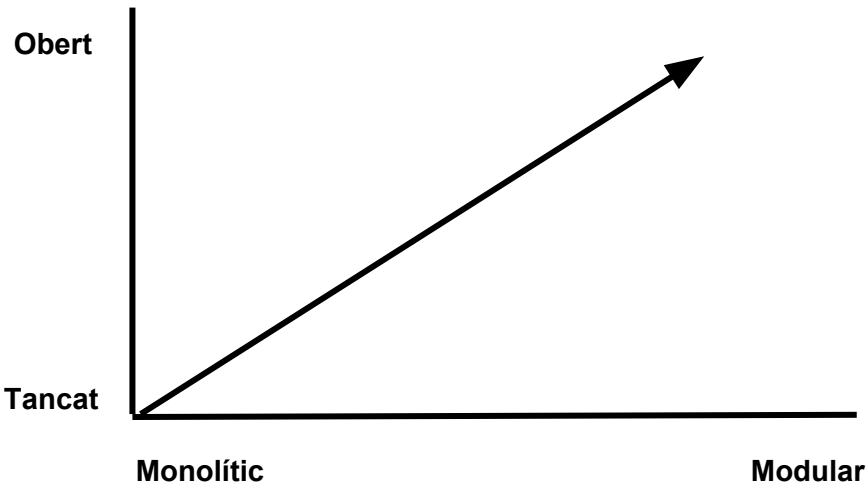
- Proveïdor/a de servidor
- Proveïdor/a de serveis
- Administrador/a de servidor
- Administrador/a de serveis
- Dissenyador
- Desenvolupador/a
- Comunitat de desenvolupament
- Responsable de suport
- Usuari/ària
- Comunitat d'usuàries

ECOSISTEMA DE PLATAFORMA - REPERCUSSIONS EN DIFERENTS EIXOS

- Accessibilitat
- Privacitat
- Sostenibilitat
- Drets laborals
- Procomú

COM DE COOPERATIVA ÉS LA PLATAFORMA?

La participació en el desenvolupament i la governança de la plataforma com a tal es debat en funció de dos eixos derivats del seu desenvolupament.



Potenciadors

- Bona qualitat de la documentació
- Dinamització de la comunitat
- Eines adequades per a cada tasca

FINESTRES DE PROCOMÚ

Espais on la comunitat pot realitzar aportacions a la plataforma en pro del bé comú

- Aportacions al codi
- Traduccions
- Fòrum de noves funcionalitats
- Fòrum de suport
- Documentació per a usuàries
- Documentació tècnica

Reutilitzar el codi

No cal que tot el codi que fa servir el nostre producte estigui escrit per nosaltres. La majoria d'aspectes que cal implementar en una aplicació web ja han estat resolts de manera eficient i publicats sota llicències lliures. Reinventar-ho no tindria sentit, no hi afegiria valor i segurament contindria més errors de funcionament i rendiment que les solucions ja establertes. És més útil i beneficiós centrar esforços i no reinventar la roda.

Git

Per poder reutilitzar codi i fer codi de manera col·laborativa cal emmagatzemar-lo fora del nostre ordinador i fer-lo accessible a la resta de membres de l'equip. Git permet la gestió de canvis en arxius i directoris al llarg del temps, de manera que podem recuperar el codi en l'estat en què es trobava en un moment concret.

En la nomenclatura de Git, la fotografia de l'estat actual del codi rep el nom de **"commit"**. Aquestes fotografies es van acumulant en el que s'anomena **"branca"**, un conjunt de commits concret i n'hi pot haver diverses. El **repositori** és el conjunt de branques que conté tot el codi i informacions del nostre projecte de programari.

Metodologies àgils

El cooperativisme de plataforma es refereix a projectes que tenen un fort vincle amb les comunitats en les quals es generen. A l'hora de dissenyar un projecte tecnològic cal tenir en compte tots els actors implicats: usuàries, persones amb perfils tècnics i tota la comunitat.

Els processos de co-disseny solen utilitzar metodologies àgils per incloure les necessitats d'una comunitat en els processos de desenvolupament i traduir-les en iteracions compostes per tasques concretes i fórmules per mesurar els resultats. **A cada iteració** s'implementa un petit grup de canvis, que aporti valor a la comunitat i que sigui fàcil d'avaluar. La **fase d'avaluació de cada iteració** alimenta les següents iteracions per incorporar les correccions i adaptacions que hagin sorgit. D'aquesta manera és més fàcil integrar la comunitat en el co-disseny perquè només cal avaluar petits canvis i comparar-los amb la visió global del projecte.

Abordar grans canvis intentant aportar el màxim valor possible en una única iteració molt llarga pot, en canvi, resultar molt difícil d'avaluar per un grup de persones nombrós.

Àgil i obert

Les metodologies àgils ja aporten un enorme valor per si soles, i per això han tingut un gran èxit en el món tecnològic empresarial no limitat a les petites i mitjanes empreses.

La combinació entre aquestes metodologies i el desenvolupament en obert aporta un potencial molt més gran en el món del cooperativisme de plataforma.

Dividir el desenvolupament del producte en tasques d'abast molt reduït i definit, com a part d'iteracions englobades en el full de ruta de la plataforma permet tant desenvolupar el producte entre vàries persones fins i tot en les mateixes àrees del programari, com un desenvolupament veloç amb molts canvis incrementals i freqüents.

Les tasques petites tenen a més el benefici afegit de no requerir un esforç cognitiu excessiu per a cap de les parts implicades. El desenvolupador entén perfectament quins són els requeriments de la tasca i compleix amb les expectatives sense haver de preguntar a la persona que ha creat la tasca. Per altra banda, la persona que s'encarrega de provar la implementació entén ràpidament quines són les parts afectades i quines funcionalitats s'han de provar.

Veure com el producte evoluciona positivament i s'avança en els objectius marcats, i com el treball en equip és efectiu i àgil té un impacte positiu en l'estat d'ànim dels membres de l'equip de desenvolupament i la comunitat en general.

Tasques d'abast reduït i ben definit són imprescindibles per obrir la porta a contribucions de membres de la comunitat i a contribucions esporàdiques de gent fora de la comunitat.

Un cop fet el triatge de tasques fàcils per nous contribuïdors cal encoratjar a la gent a participar en el desenvolupament, no només en codi. Participar de manera activa requereix reconèixer les mancances pròpies, deslligar-se de les pròpies vergonyes i centrar-se en les potencialitats que es poden aportar al projecte.

Quan això va seguit d'un acompanyament augmenten molt les probabilitats que de ser un contribuïdor esporàdic es passi a ser un membre més de la comunitat. Això requereix esforç i dedicació dels membres més actius i de l'equip de desenvolupament.

Operacions i administració de sistemes

Un dels aspectes menys evidents en tot el procés de producció de programari és "posar a producció" el mateix software, posar-lo a disposició dels usuaris. En funció de l'arquitectura de xarxa i de l'aplicació que s'hagi escollit hi ha diverses maneres de portar a terme aquest procés.

En el procés client/servidor, l'equip responsable d'entregar el producte als seus usuaris necessita traslladar el programari a servidors accessibles des d'internet. Els processos necessaris per fer-ho estan estretament relacionats amb el conjunt de tecnologies escollides per donar suport al programari produït. Tot i aquesta especificitat, hi ha una sèrie de patrons recurrents en aquests processos. Cal adoptar pràctiques que permetin que tot l'equip conegui aquests processos de manera que no depenguin de persones concretes.

Els processos bàsics necessaris per posar un programari a producció es poden resumir en:

* Gestió de servidors
* Provisionament
* Desplegament
* Monitoratge

Gestió de servidors

L'elecció dels servidors en els quals s'executarà el programari requereix coneixement del mercat i dels factors relacionats amb el posicionament geogràfic dels usuaris del programari. Hi ha molts proveïdors de serveis d'allotjament, per això cal entendre també el nivell de confiança que podem esperar segons el model de governança, la mida de l'empresa, etc.

Aprovisionament ("Provisioning")

Una vegada aconseguit accés als servidors, cal preparar les màquines perquè puguin rebre i executar el nostre programari. El procés d'aprovisionament s'encarrega d'instal·lar en els servidors tot aquell programari que no hem desenvolupat nosaltres directament però que necessitem per executar el nostre. Un exemple pot ser el programari que gestiona les bases de dades. Aquest programari se sol anomenar "dependència".

Desplegament

A partir del moment en què els servidors estan equipats amb totes les dependències necessàries podem començar el procés de trasllat del programari produït pel nostre equip. Aquest procés és responsable de posar físicament a producció el programari. Un aspecte important d'aquest procés és que ha de facilitar l'accés a la versió del programari que ha estat desplegada i a processos d'emergència per tornar a activar versions anteriors.

Monitoratge

Tot el conjunt de programari i processos necessaris per mantenir accessible el nostre programari pot resultar difícil de mantenir i, sobretot, difícil de gestionar en casos d'emergències. Per reduir aquesta amenaça se solen engegar mecanismes i processos de monitoratge. Es tracta d'instrumentar i monitorar el nombre més gran d'aspectes possible, partint del correcte funcionament del servidor (memòria, CPU, etc.) fins a cada peça del nostre conjunt de programari que resulta vital per al nostre projecte.

Obtenint totes aquestes dades es pot engegar un sistema d'alertes que avisi l'equip cada cop que alguns valors mesurats s'allunyin dels valors establerts. Aquestes dades i informacions també són importants a l'hora d'analitzar un problema ocorregut i per actuar perquè no es torni a repetir.

Processos compartits

Acabem d'entendre la complexitat i alhora pes fonamental que tenen els processos de posada a producció d'un programari, una fallada en aquest conjunt pot posar en risc tot el treball invertit en el desenvolupament del projecte. Per aquesta raó és recomanable adoptar bones pràctiques i compartir el coneixement entre projectes que usen tecnologies semblants.

Documentar-ho tot

Centralitzar el coneixement pot posar en perill un projecte tecnològic d'aquest tipus. Cal contrarestar qualsevol tipus de resistència a compartir la informació encara que pugui resultar en una perduda de poder per a algunes persones. És important documentar cada decisió i cada canvi en cadascun dels processos esmentats. Si per alguna raó hi ha parts d'aquests processos que s'han implementat sense generar documentació cal aprofitar l'ocasió per demanar a una persona que mai hagi tingut accés a aquests processos de documentar-los. La millor persona per documentar un procés és la que mai els hagi posat en pràctica, és una bona ocasió per transmetre coneixement i alhora generar documentació clara i entenedora.

Reutilitzar processos entre projectes

En els casos en els quals aquesta cultura de compartir coneixement no hagi estat adoptada per tot l'equip, ens podem trobar amb processos que solucionen problemes de forma ineficient. Generalment és a causa de persones que desconeixien o ignoraven l'existència de processos semblants i a la falta de comunicació. És important reutilitzar els processos entre els projectes i trobar la forma d'implicar totes les persones interessades quan es treballa en la implementació de nous processos.

Fàcil replicació

Des d'una perspectiva del pro-comú digital tot aquest treball i esforç dedicat a documentar i compartir els processos de producció s'ha de fer visible també fora de la mateixa organització publicant tot el que es pugui amb llicències obertes (Creative Commons, etc.). D'aquesta manera es podrà potenciar que els projectes es repliquin, ja que, com hem vist, no solament el programari en si és important i estratègic sinó que també ho són tot el conjunt de processos per a la posada a producció i el manteniment del projecte. Exemple: https://github.com/loomio/loomio-coop-handbook

RECURSOS ÚTILS			
Títol i accés	**Utilitat?**		
	*	**	***
Building on Blockchain (Resonate) https://resonate.is/building-on-blockchain/			
Intro to Crypto-economics (Peter Harris) https://medium.com/@peteratomic/intro-to-crypto-economics-9508e471d617			
Loomio Cooperative Handbook https://github.com/loomio/loomio-coop-handbook			
What is co-design (Design for Europe) http://designforeurope.eu/what-co-design			
Design Sprint Kit (Google) https://designsprintkit.withgoogle.com/			
Triage issues in 7 simple steps (GitLab) https://about.gitlab.com/2017/10/26/triage-issues-gitmate/			
Using Loomio to govern a self-organising community (Enspiral) https://medium.com/enspiral-tales/using-loomio-to-govern-a-self-organising-community-45ef4af15f26			
Blockchain Doesn't Decentralise Power… Unless You Design It To (Enspiral) https://medium.com/enspiral-tales/blockchain-doesnt-decentralise-power-5918c168e6f6			