# El dia a dia d'un projecte àgil: Scrum. Des de la conceptualització al lliurament del producte

Marcos Bermejo
Marc Florit
Ramon G. Sedó

PID_00236239

Temps de lectura i comprensió: **2 hores**

UOC
Universitat Oberta de Catalunya

# Índex

# 1. Introducció a Scrum

Scrum és un marc de treball per a definir processos que es caracteritza perquè és lleuger i fàcil d'entendre.

Scrum no defineix un procés concret, sinó que proporciona les eines perquè cada equip adapti el marc de treball i trobi el procés més adequat a les seves circumstàncies.

Aquesta característica fa que Scrum sigui apropiat per a entorns complexos en què els resultats són *a priori* incerts, en els quals la tecnologia a utilitzar no sigui trivial i fins i tot desconeguda i on, és difícil predir el que passarà en el futur i en què, per tant, caldrà una gran capacitat d'adaptació com en el cas del desenvolupament de productes multimèdia.

## 1.1. Fonaments de Scrum

### 1.1.1. Gestió empírica

Scrum és un **mètode empíric** i, per tant, es basa a gestionar el procés de desenvolupament a partir de l'experiència (observació) i no pas per mitjà de prediccions. La conseqüència principal d'això és que les decisions es prenen tenint en compte fets coneguts, en comptes de fets hipotètics.

Per entendre la diferència entre l'enfocament empíric i l'enfocament predictiu, suposem que ens encarreguen un projecte que, *a priori*, estimem que tindrà un any de durada.

Si gestionem el procés d'una manera predictiva, farem una predicció de les tasques necessàries i de l'esforç que haurem de dedicar a cada tasca i, a partir d'això, establirem un calendari. Durant l'execució del projecte compararem la situació real amb la predicció i intentarem ajustar la realitat (per exemple, afegint més recursos) perquè s'adeqüi al calendari establert.

Amb Scrum, en canvi, establirem igualment una predicció inicial però durant l'execució del projecte ens anirem qüestionant, d'una manera regular, la correcció de la predicció, i l'ajustarem (per exemple, adaptarem el calendari o modificarem el conjunt de funcionalitats) perquè s'adeqüi a la realitat.

### 1.1.2. Cicle de vida iteratiu i incremental

Per a poder adaptar el procés a la realitat, Scrum utilitza el **cicle de vida iteratiu i incremental**. És a dir, organitzem el projecte en iteracions curtes (entre una i quatre setmanes, preferentment) per tal de tenir un ritme estable per a la revisió del procés i la identificació d'oportunitats de millora.

Tanmateix, el producte es construeix d'una manera incremental, amb la qual cosa s'aconsegueix un objectiu doble:

**1)** Des d'etapes molt primerenques tenim implementades les funcionalitats més importants del projecte.

**2)** Al mateix temps, les diferents iteracions són comparables entre si (cosa que no passa, per exemple, en el cicle de vida en cascada, en què cada etapa és totalment diferent de les altres i, per tant, no podem acumular tant coneixement d'una etapa a la següent), per la qual cosa podem obtenir mètriques objectives del ritme sostingut al que anem lliurant producte, i per tant valor al client.

### 1.1.3. Transparència

Tots els aspectes del procés han de ser visibles en tot moment tant als responsables del projecte com a l'equip que l'està construint. La transparència requereix la definició d'un criteri comú per a tots els observadors, de manera que interpretin el que veuen de la mateixa manera.

> Per exemple, és molt important que totes les persones implicades en el projecte comparteixin la mateixa "definició de fet", és a dir, que tothom coincideixi a l'hora de valorar si una tasca està acabada. És habitual que diverses persones tinguin idees diferents sobre aquesta definició, ja que, per exemple, potser un desenvolupador considera que una funcionalitat està feta quan ha acabat la seva part, però el responsable del producte considera que no ho està fins que no s'ha integrat amb la resta del sistema.

### 1.1.4. Inspecció i adaptació

Es dóna molta importància a la inspecció freqüent del procés i dels resultats que s'obtenen, com també a l'adaptació del procés quan els resultats es desvien del que era previsible, de manera que el resultat final no sigui acceptable.

En concret, Scrum ens proposa quatre moments per a inspeccionar el procés i fer propostes d'adaptació:

**1)** Quan es planifica l'iteració (cada una, dues, tres o quatre setmanes depenent de la durada de la iteració i de cada equip).

**2)** En la reunió diària (en què es sincronitza l'estat real de cada tasca i es comparteixen els problemes que es van trobant).

**3)** Quan es revisa i es mostra la feina feta al final de cada iteració.

**4)** En la retrospectiva que es fa al final de cada iteració (de fet, la finalitat de la retrospectiva és, precisament, reflexionar sobre com ha anat la iteració i decidir quines millores cal aplicar).

### 1.2. Beneficis de Scrum

Un dels **objectius de Scrum** és obtenir el màxim valor possible de l'esforç dedicat al desenvolupament de productes multimèdia.

Mitjançant el **desenvolupament incremental**, en què es desenvolupa el producte multimèdia a base d'afegir funcionalitats completes al producte existent, i la **priorització**, segons el valor que la funcionalitat ofereix al client, aconseguim que el client obtingui molt aviat un producte amb les principals funcionalitats implementades d'una manera completa i, al mateix temps, estalviem –si ho volem– haver d'implementar les funcionalitats que aporten menys valor, amb l'estalvi de costos consegüent.

Tanmateix, Scrum facilita la **transparència i la visibilitat dels problemes**. Mitjançant la reflexió constant sobre el procés i la identificació d'impediments al més aviat possible (fins al punt que se'n parla diàriament), ens ajuda a identificar els problemes abans que siguin massa grans i, per tant, contribueix a mantenir un nivell més alt de productivitat i qualitat.

Per exemple, si un equip té un problema perquè els desenvolupadors no poden avançar atès que el representant del client no està disponible per a solucionar dubtes sobre els requisits, és molt millor detectar-ho al principi de la iteració que no pas esperar que es comencin a no complir les dates previstes i es porti a terme una anàlisi del problema.

Un altre avantatge de Scrum és que **minimitza la necessitat de gestió**, ja que dóna als equips de desenvolupament l'autonomia necessària per a organitzar-se per mitjà de la comunicació i el consens. La direcció de l'empresa estableix les prioritats i l'equip de desenvolupament decideix la manera de satisfer-les. Això facilita la feina dels gestors (que només han de decidir les prioritats, però no han d'assignar les tasques) i augmenta la satisfacció i el compromís dels desenvolupadors, ja que gaudeixen d'una certa autonomia.

Quan comença el cicle de desenvolupament en Scrum (la iteració), el responsable del producte decideix quines funcionalitats s'han d'implementar en aquest cicle. Un cop decidida la llista de funcionalitats, els membres de l'equip de desenvolupament poden escollir, del conjunt de tasques necessàries per a implementar totes aquestes funcionalitats, en quina volen treballar en un moment donat.

Finalment, com que Scrum és un procés molt orientat a la millora contínua i a l'adaptació, tant del procés mateix com del producte desenvolupat, és més fàcil aconseguir productes que satisfacin els clients i que siguin innovadors.

# 2. Com funciona Scrum

**1) El *product backlog***

Un projecte de Scrum és impulsat per una visió de producte elaborada pel propietari del producte, i s'expressa a la **pila de producte**. El *product backlog* és una llista prioritzada del que es requereix, per ordre de valor per al client o negoci, amb els elements de més valor a la part superior de la llista. El *product backlog* evoluciona durant la vida útil del projecte, i els elements s'afegeixen, es treuen o són reprioritzats contínuament.

**2) L'*sprint***

Scrum estructura el desenvolupament de productes en cicles de treball anomenats *sprints*, repeticions de treball que típicament tenen d'una a quatre setmanes de durada. Els *sprints* són de durada determinada; mai no s'estenen més enllà de la data fixada al principi, independentment que la feina planificada per l'*sprint* s'hagi completat o no.

**3) Planificació de l'*sprint***

Al començament de cada *sprint*, es porta a terme la **reunió de planificació de l'*sprint***. El propietari del producte i l'equip Scrum (amb la facilitació de l'Scrum màster) revisen la **pila de producte**, discuteixen els objectius i el context de les històries d'usuari, i l'equip Scrum selecciona els elements de la pila de producte que es comprometen a realitzar per al final de l'sprint, partint de la part superior de la pila de producte, on es troben els elements de priorització més alta.

Cada element seleccionat a la pila de producte es descompon després en una sèrie de tasques individuals. La llista de tasques es registra en un document anomenat ***sprint backlog***.

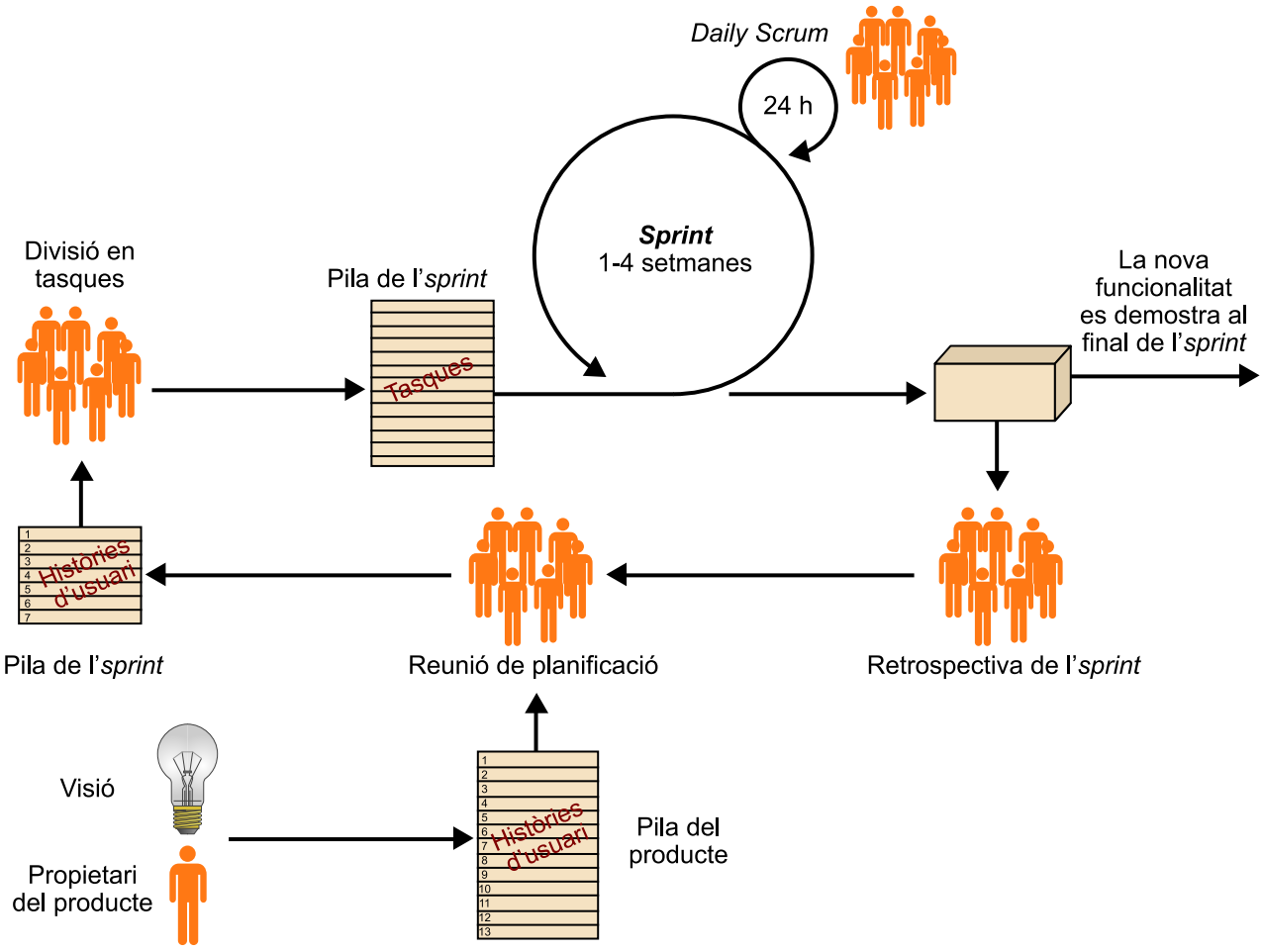
**4) Reunió diària de Scrum**

Una vegada l'*sprint* ha començat, l'equip Scrum es dedica a una altra de les pràctiques principals de Scrum: la **reunió diària de Scrum** o ***daily stand-up*** en la denominació original en anglès. Es tracta d'una trobada breu (quinze minuts, com a màxim) que es duu a terme cada dia de feina a una hora determinada. Tots els membres de l'equip hi assisteixen. En aquesta reunió, es presenta la informació necessària per a inspeccionar l'avenç des del dia anterior. Aquesta informació pot resultar en la replanificació i en debats sobre la funcionalitat després del *daily stand-up*.

### 5) Revisió de l'*sprint* i retrospectiva

Un cop finalitzat l'*sprint*, es duu a terme la reunió de revisió de l'*sprint*, en què l'equip Scrum i les parts interessades inspeccionen el que es va fer durant l'*sprint* i ho discuteixen. En aquesta reunió, hi són presents el propietari del producte, els membres de l'equip i l'Scrum màster, a més dels clients, gent de negoci, experts, executius i qualsevol altra persona interessada.

Després de la revisió de l'*sprint*, l'equip es reuneix per a la retrospectiva de l'*sprint*, una oportunitat perquè l'equip discuteixi el que ha funcionat bé durant l'*sprint* i el que no funciona, i es posin d'acord en els canvis que cal intentar fer en l'*sprint* següent per tal de ser més productius.



# 3. Els rols en Scrum

L'equip Scrum es compon de tres rols clarament diferenciats:

**1) El propietari del producte**. Pren les entrades del que ha de ser el producte i ho tradueix en una visió de producte.

**2) L'equip**. Desenvolupa el producte previst pel propietari del producte.

**3) L'Scrum màster**. Té tot el que es necessita perquè l'equip Scrum assoleixi l'èxit. Això passa per eliminar els obstacles d'organització, facilitar les reunions i actuar com un guardià perquè ningú no obstaculitzi la feina de l'equip.

## 3.1. El propietari del producte

El **propietari del producte** és responsable de maximitzar el retorn sobre la inversió (ROI, *return on investment*) mitjançant la identificació de les característiques del producte i la traducció d'aquestes en una llista de característiques de prioritats, decidir què ha de figurar a la part superior de la llista per a l'*sprint* següent i reprioritzar i refinar la llista contínuament.

El propietari del producte té la responsabilitat de pèrdues i guanys per al producte, suposant que és un producte comercial. En el cas d'una aplicació interna, el propietari del producte no és responsable del retorn de la inversió en el sentit d'un producte comercial (que generarà ingressos), però continua sent responsable de maximitzar el retorn de la inversió en el sentit d'escollir –cada *sprint*– els elements que tenen més valor de negoci i un cost més baix.

El propietari del producte no és un gerent de producte.

En alguns casos, el propietari del producte i el client són la mateixa persona, fet comú per a les aplicacions internes. En altres casos, el client pot ser milions de persones amb una varietat de necessitats, i aleshores la funció de propietari del producte és similar a la d'un responsable de producte o de màrqueting del producte en moltes organitzacions. No obstant això, el propietari del producte és una mica diferent del tradicional gerent de producte, ja que activa i sovint interacciona amb l'equip, personalment, oferint les prioritats i revisant els resultats de cada iteració de dues a quatre setmanes, en comptes de delegar les decisions de desenvolupament a un gerent de projecte. És important assenyalar que en Scrum hi ha una sola persona que exerceix de propietari

del producte i té l'autoritat final. En els programes de diversos equips, aquest propietari del producte pot delegar la feina als propietaris de productes que el representen en els equips de subordinats, però totes les decisions i la direcció provenen del nivell més alt, del propietari únic del producte.

## 3.2. L'equip

L'**equip** construeix el producte del qual el client farà ús: l'aplicació o pàgina web, per exemple. L'equip en Scrum és multifuncional i inclou tots els coneixements necessaris per a lliurar el producte **potencialment lliurable** en cada *sprint*. També és autoorganitzat, de manera que l'equip s'autogestiona amb un alt grau d'autonomia.

Per tant, no hi ha cap d'equip o cap de projecte en Scrum. En canvi, els membres de l'equip decideixen a què es comprometen i quina és la millor manera de complir amb aquest compromís. L'equip és **autoorganitzat**.

### 3.2.1. Equip dedicat

L'equip en Scrum és de set més o menys dues persones, és a dir, un mínim de cinc persones un màxim de nou. Per a un producte multimèdia l'equip pot incloure programadors, dissenyadors d'interfície i els provadors. L'equip desenvolupa el producte i ofereix idees per al propietari del producte sobre la manera de fer un gran producte. És essencial que l'equip estigui completament dedicat a treballar per a un únic producte durant l'*sprint*, ja que, si no és així, la multitasca provocada per la col·laboració en múltiples productes o projectes en limitarà seriosament el rendiment. Equips estables s'associen amb més productivitat, de manera que canviar els membres de l'equip també es una cosa que s'ha d'evitar.

Els grups d'aplicacions amb moltes persones s'organitzen en diversos equips Scrum, cadascun enfocat a les diferents característiques del producte, amb una estreta coordinació dels seus esforços. Des d'un equip es fa tota la feina (planificació, anàlisi, programació i prova) per a una funció completa centrada en el client, per la qual cosa els equips de Scrum també es coneixen com a **equips de característiques o funcionalitats**. En els programes tècnicament molt complexos, hi pot haver equips organitzats per nivell arquitectònic, com quan s'empra l'arquitectura d'una família de productes formada per més d'un producte. No obstant això, la integració abans del final de l'*sprint* és més difícil quan els equips estan tan estructurats.

### 3.3. L'Scrum màster

L'Scrum **màster** ajuda el grup del producte a aprendre i aplicar Scrum per a aconseguir valor de negoci. L'Scrum màster és el que té la capacitat d'ajudar l'equip a tenir èxit.

L'Scrum màster no és el mànager de l'equip o un director de projecte, sinó que serveix a l'equip, el protegeix de la interferència externa, i educa i orienta el propietari del producte i l'equip en l'ús de Scrum. L'Scrum màster assegura que tots en l'equip (incloent-hi el propietari del producte, i els de gestió) entenguin i segueixin les pràctiques de Scrum. També contribueix a dirigir l'organització per mitjà del canvi, una activitat sovint difícil però necessària per a aconseguir l'èxit amb el desenvolupament àgil.

#### 3.3.1. El compromís és important

Com que Scrum fa visibles molts impediments i amenaces a l'eficàcia de l'equip i del propietari del producte, és important tenir un Scrum màster compromès que treballi enèrgicament per ajudar a resoldre aquestes qüestions. Si no és així, serà més difícil per a l'equip o el propietari del producte tenir èxit. Els equips de Scrum han de tenir una dedicació a temps complet d'un Scrum màster, per bé que un equip més petit pot tenir un membre de l'equip exercint aquest paper (que comporta una càrrega més lleugera de treball regular, quan ho fan). Els Scrum màsters poden provenir de qualsevol fons o disciplina: enginyeria, disseny, proves, gestió de producte i gestió de projectes o de qualitat.

L'Scrum màster i el propietari del producte no poden ser la mateixa persona, ja que, de vegades, l'Scrum màster pot ser cridat per a fer retrocedir el propietari del producte (per exemple, si el propietari tracta d'introduir nous productes a la meitat d'un *sprint*). A diferència d'un director de projecte, l'Scrum màster no diu a les persones el que han de fer ni assigna tasques. Si l'Scrum màster estava prèviament en una posició de gestió de l'equip, haurà de canviar molt la manera que té de pensar i l'estil d'interacció amb l'equip per a tenir èxit amb Scrum. En cas que un gestor faci la transició a la funció de Scrum màster, el millor és fer servir un equip diferent del que treballava anteriorment com a gestor, a fi d'evitar possibles conflictes.

**Què passa amb els gerents?**

Tingueu en compte que no hi ha el paper de cap de projecte en Scrum. De vegades, un (ex) cap de projecte pot entrar en el rol de Scrum màster, però això té un historial d'èxits i fracassos –hi ha una diferència fonamental entre els dos papers, tant en les responsabilitats del dia a dia com en la mentalitat necessària per a assolir l'èxit.

A més dels tres rols principals, hi ha altres col·laboradors per a assolir l'èxit del producte, incloent-hi els directius. Malgrat que els seus rols canvien, continuen sent valuosos, per exemple, per als aspectes següents:

- Crear un model de negoci que funciona i proporcionar els recursos que l'equip necessita.
- Donar suport a l'equip, respectant les regles i l'esperit de Scrum.
- Ajudar a eliminar els obstacles que l'equip identifica.
- Posar els seus coneixements i experiència a disposició de l'equip.
- Desafiar l'equip a anar més enllà de la mediocritat.

En Scrum, aquests individus reemplacen el temps que abans es dedicava a exercir el paper de "mainadera" (assignació de tasques, obtenció d'informes d'estat i altres formes de microgestió) amb el temps com a *guru* i *servent* de l'equip (*mentoring*, *coaching*, ajudar a eliminar obstacles, contribuir a resoldre problemes, fer aportacions creatives i guiar el desenvolupament de les competències dels membres de l'equip). En aquest canvi, els gerents poden necessitar modificar l'estil de gestió, per exemple, mitjançant l'interrogatori socràtic, per a ajudar l'equip a trobar la solució a un problema, en comptes de decidir una solució i implementar-la a l'equip.

# 4. Primers passos

Iniciar un projecte Scrum no és difícil, sempre que es facin els passos a l'hora i s'asseguri que tothom s'hi sent inclòs.

## 4.1. El *product backlog* o pila de producte

Un projecte de Scrum és impulsat per una visió de producte elaborada pel propietari del producte, i s'expressa a la **pila de producte**. El *product backlog* és una llista prioritzada del que es requereix, per ordre de valor per al client o negoci, amb els elements de més valor a la part superior de la llista. El *product backlog* evoluciona durant la vida útil del projecte, i els elements s'afegeixen, es treuen o són reprioritzats contínuament.

Aquest registre no solament existeix i evoluciona al llarg de la vida útil del producte, sinó que a més és el full de ruta del producte.

En qualsevol moment, la **pila de producte** és el punt de vista únic i definitiu de "tot el que es pot fer per l'equip, en ordre de prioritat". Només hi ha una única pila de producte.

La pila de producte inclou una varietat d'elements, sobretot les noves característiques del producte ("permetre als usuaris col·locar el llibre a la cistella de la compra"), però també els objectius de millora d'enginyeria ("revisar el mòdul de processament de transaccions perquè sigui escalable"), d'exploració o treball de recerca ("investigar solucions per a accelerar la validació de la targeta de crèdit"), els requisits de rendiment i seguretat, i, possiblement, els defectes coneguts ("diagnosticar i corregir els errors de processament de comandes de seqüència"), si només hi ha uns quants problemes (un sistema amb molts defectes en general té un sistema de seguiment de defectes de manera separada).

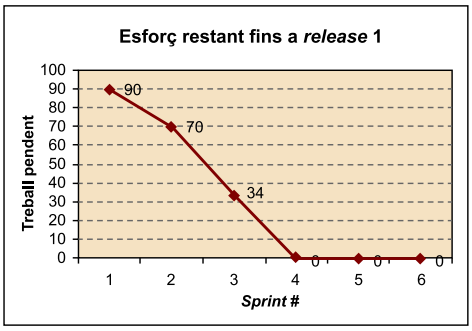
Generalment, s'articulen els requisits en termes d'**històries d'usuari**: descripcions concises i clares de la funcionalitat en termes del valor que tenen per a l'usuari final del producte.

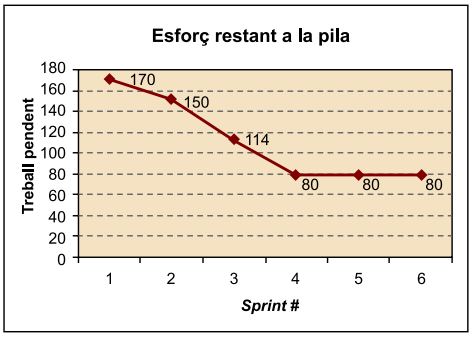
El subconjunt de la pila de producte que es destina per a la versió actual es coneix com a **pila de la versió** (o ***release backlog*** en l'anglès original), i normalment, aquesta part és l'objectiu principal del propietari del producte.

**Aplicació de climatologia per dispositius mòbils**

| ID | Descripció | *Sprint* # | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| | | Esforç requerit per la primera versió a l'inici de l'*sprint* | 90 | 70 | 34 | 0 | 0 | 0 |
| 1 | Muntar el sistema d'integració contínua | | 5 | 0 | 0 | 0 | 0 | 0 |
| 2 | Crear l'esquelet de l'aplicació | | 5 | 0 | 0 | 0 | 0 | 0 |
| 3 | Mostrar la temperatura actual de la manera més simple | | 13 | 0 | 0 | 0 | 0 | 0 |
| 4 | Muntar el servidor per a obtenir dades de temperatura | | 3 | 0 | 0 | 0 | 0 | 0 |
| 5 | Implementar suport del protocol WeatherML en el servidor | | 13 | 0 | 0 | 0 | 0 | 0 |
| *Sprint* 1 | *Obtenir dades de prova des del servidor fins al client* | | | | | | | |
| 6 | Suport de gràfics al costat client | | 20 | 0 | 0 | 0 | 0 | 0 |
| 16 | Dibuixar amb la llibreria de gràfics una icona i un text de temperatura al client | | - | 13 | 0 | 0 | 0 | 0 |
| 17 | Dibuixar la pantalla de temperatura real | | - | 8 | 0 | 0 | 0 | 0 |
| 7 | Implementar el suport per a múltiples dies | | 8 | 8 | 0 | 0 | 0 | 0 |
| 8 | Implementar el suport per pluja o neu | | 2 | 2 | 0 | 0 | 0 | 0 |
| 9 | Suportar el camí de ciutat | | - | 5 | 0 | 0 | 0 | 0 |
| *Sprint* 2 | *Versió mínima funcionant* | | | | | | | |
| 10 | Capturar la temperatura d'un dia del proveïdor de climatologia | | ? | 13 | 13 | 0 | 0 | 0 |
| 11 | Obtenir dades de pluja, neu, etc. del proveïdor | | 8 | 8 | 8 | 0 | 0 | 0 |
| 12 | Obtenir dades per a més d'un dia | | 5 | 5 | 5 | 0 | 0 | 0 |
| 13 | Refresc automàtic al client | | 8 | 8 | 8 | 0 | 0 | 0 |
| *Sprint* 3 | *Versió amb dades de temperatures reals* | | | | | | | |
| ***Release* 1** | **Versió per a vendre al *marketplace*** | | | | | | | |
| 14 | Injectar *ads* simulats des del servidor | | 20 | 20 | 20 | 20 | 20 | 20 |
| 15 | Obtenir i mostrar *ads* reals | | 20 | 20 | 20 | 20 | 20 | 20 |
| 16 | Canviar de ciutat automàticament a partir de les dades del mòbil | | 40 | 40 | 40 | 40 | 40 | 40 |
| *Sprint* 4 | *Suport per ads publicitaris (monetització)* | | | | | | | |
| ***Release* 2** | **Versió amb publicitat** | | | | | | | |
| | | **Esforç total a la pila del producte** | 170 | 150 | 114 | 80 | 80 | 80 |

**Estat del *backlog* després de l'*sprint*** 3 **Versió** 1




Exemple d'una pila de producte

La pila de producte s'actualitza constantment pel propietari del producte per a reflectir els canvis en les necessitats del client, les noves idees o punts de vista, els moviments de la competència, obstacles tècnics que apareixen, entre d'altres. L'equip ofereix al propietari del producte les estimacions de l'esforç requerit per a cada element de la pila de producte. A més, el propietari del producte és responsable d'assignar una estimació del valor de negoci a cada element individual. Això sol ser una pràctica desconeguda per a un propietari del producte novell.

Com a tal, un Scrum màster pot ajudar el propietari del producte a aprendre a fer-ho. Amb aquestes dues estimacions (**esforç** i **valor**) i potser amb les estimacions de risc addicionals, el propietari del producte dóna prioritat a la cartera de comandes per maximitzar el retorn de la inversió (triant elements d'alt valor amb poc esforç) o, en segon lloc, per reduir algun risc important. Com es veurà, aquestes estimacions d'esforç i valor es poden actualitzar en cada *sprint*, ja que les persones aprenen i, en conseqüència, es produeix una reprioritització contínua de l'activitat i la pila de producte evoluciona constantment.

Scrum no obliga a realitzar les estimacions de la pila de producte d'una manera determinada, però és habitual l'ús d'estimacions relatives, qualificades com a **punts d'història** en comptes d'unitats absolutes d'esforç, com a hores persona. Amb el temps, i un bon seguiment dels resultats de l'equip, es poden treure conclusions a partir de la quantitat de punts d'història relatius que un equip de producció multimèdia és capaç d'implementar per cada *sprint* (per exemple, una mitjana de 26 punts d'història per *sprint*). Amb aquesta informació es pot projectar una data de llançament per a completar totes les característiques,

o bé el nombre de característiques que es completaran probablement abans d'una data. Dels punts completats per l'*sprint*, se'n diu **velocitat de l'equip**. Un pla de llançament realista sempre es basa en la velocitat de l'equip.

Els elements de la pila de producte poden variar significativament en mida i esforç. Els més grans es fragmenten en elements més petits, normalment durant reunions de refinament de la pila del producte o bé en la reunió de planificació de l'*sprint*.

**La quantitat de detalls?**

Un dels mites sobre Scrum és que impedeix escriure les especificacions detallades. En realitat, el propietari del producte i l'equip són els que han de decidir quant de detall es requereix, i això pot variar d'un element a un altre, depenent de la visió de l'equip i d'altres factors. S'ha de procurar reflectir el que és important en la menor quantitat d'espai necessari –en altres paraules, no descriure tots els detalls possibles d'un element, simplement deixar clar el que cal perquè s'entengui. Els elements de baixa prioritat, que són propensos a ser implementats en una etapa posterior, en general tindran menys detalls respecte als requisits d'alta prioritat, que aviat es duran a terme i per tant tendeixen a tenir més detalls.

### 4.1.1. La visió del producte: pensant en l'usuari. Les històries d'usuari

La llista de tasques definides a la pila de producte, tot i que SCRUM no ho defineix, sorgeixen en la majoria d'ocasions de les històries d'usuari, un dels formats més utilitzats per a crear les piles.

Una **història d'usuari** és una redacció breu d'una funcionalitat que per si mateixa dóna valor a l'usuari.

Es compon d'un títol, una descripció breu sobre el que ha de fer la funcionalitat i unes condicions d'acceptació. Opcionalment, es poden afegir anotacions i acotacions a la història.

És molt aconsellable fer servir l'estructura següent:

Com a [rol d'usuari] vull [funcionalitat] per a [objectiu].

**Com ha de ser una història d'usuari**

Una bona història d'usuari segueix les sigles **INVEST**, és a dir, és:

- **I**ndependent.
- **N**egociable.
- **V**aluosa per als usuaris i clients.
- **E**stimable (es pot mesurar).
- ***S**mall* (suficientment petita per a poder-hi treballar).

- ***T**estable* ('verificable').

**1) Independent**

La dependència entre històries a vegades deriva en problemes en la planificació i la priorització. Si una història molt poc prioritària està relacionada amb una altra que ho és molt, la priorització serà complicada.

Quan dues històries estan relacionades, és molt difícil fer l'estimació, ja que sorgeixen dubtes com ara: "si desenvolupéssim la història A en primer lloc, la història B seria més fàcil de fer i requeriria menys d'esforç, però la història A no és prioritària". En la gestió àgil de projectes el més prioritari sempre es fa abans, amb la qual cosa entraríem en un conflicte.

Quan es presenta aquest tipus de dependència entre històries, hi ha dos camins per a solucionar-ho:

- Combinar les històries dependents en una de més gran, però independent de la resta.

- Trobar un camí diferent en la definició d'aquestes històries.

**2) Negociable**

Les històries d'usuari són descripcions curtes de funcionalitats, els detalls de les quals s'han de negociar en converses posteriors entre el client i l'equip de producció. Com hem dit al començament, una història es compon d'una descripció breu acompanyada opcionalment d'anotacions o acotacions de la història. Aquestes acotacions s'han de negociar al llarg del projecte.

**Acotar la funcionalitat**

Les anotacions d'una història han d'acotar la funcionalitat. És a dir, si tenim la funcionalitat "com a usuari administrador, vull poder generar un document Word amb la llista de noms dels productes del sistema", l'anotació "la funcionalitat també ha de treure la llista en PDF" no acota ni detalla la funcionalitat, sinó que l'augmenta. Una anotació possible seria "el document Word ha de tenir el format oficial de l'empresa".

En molts projectes, es confon "donar més detalls" amb "donar més precisió". No cal entrar en la precisió excessiva; si la història es pot estimar i prioritzar, es poden deixar els detalls per a converses posteriors.

Podem veure una història com un recordatori de tenir una conversa entre l'equip de producció i el client, en què la descripció indica la funcionalitat sobre la qual hem de parlar i les anotacions serien els detalls als quals s'han arribat en aquestes converses.

**3) Valuosa**

La millor manera d'assegurar-se que una història és valuosa per al client és deixar que aquest escrigui les històries. Habitualment, els clients es mostren incòmodes amb aquesta proposta, ja que estan acostumats que tot el que escriuen és un contracte i es pot fer servir en contra seu ("com que no vas escriure que volies validació del DNI..."). En la gestió àgil de projectes no es busca aquest "passar la pilota". És molt estrany que detalls importants no es comentin en les converses entre el client i l'equip de producció.

Si aconseguiu tenir un client que confia en vosaltres i s'implica en el projecte, teniu molt de guanyat en la definició d'històries.

**4) Estimable**

Hi ha tres raons habituals per les quals una història d'usuari no es pot estimar:

- Poc domini per part de l'equip de producció del llenguatge del client.
- Poc domini tècnic per part de l'equip de producció sobre la tecnologia que cal fer servir.
- La història és massa gran.

Si l'equip de producció no entén la història tal com s'ha escrit, la solució és realitzar reunions entre l'equip de producció i el client. En la gestió àgil de projectes no està permès estalviar-se converses ni suposar o interpretar conceptes. Si l'equip de producció no està segur del que ha de fer, cal parlar-ne per a entendre-ho.

Si el problema és tecnològic, la solució és reservar un o dos programadors per a un *spike*.

Un ***spike*** és un experiment breu per a aprendre com funciona una tecnologia, i que en el nostre projecte implicaria fer-la servir.

Durant l'*spike*, es demana als programadors que experimentin i aprenguin tot el que cal només per a estimar la funcionalitat, no que la desenvolupin.

**Nota**

És recomanable que un *spike* estigui limitat per un *time-box*, o porció de temps, en què si se sobrepassa i no s'ha aconseguit res, podem descartar la tecnologia seleccionada.

Si el problema és que la història d'usuari és massa gran per a estimar-la amb una certa precisió, l'única sortida és dividir-la (més endavant aprendrem a dividir les històries d'usuari).

**Història d'usuari petita**

Pot ser que una història d'usuari com la següent:

"Com a usuari visitant de la web, he de poder registrar-me per a entrar al sistema."

sigui suficientment petita, ja que la tecnologia emprada dóna eines prefetes per a dur a terme aquesta tasca.

Encara que una història sigui massa gran, de vegades és útil com a èpica. Les **històries èpiques** són recordatoris de les grans parts del sistema que han de ser desenvolupades.

**5) *Small***

La mida ideal de les històries varia depenent de l'equip, de les capacitats i de la tecnologia emprada.

Si la història s'ha de dividir, cal tenir en compte les consignes següents:

**a) Dividir històries per la composició que tenen**

Hem de pensar en el flux d'accions que l'usuari durà a terme en tota aquesta funcionalitat; per exemple, "com a usuari administrador vull poder administrar els productes del sistema" es podria dividir en "veure una llista de productes", "crear un producte", "eliminar un producte" o "editar un producte". Tenir presents les sigles **CRUD** (***create***, ***read***, ***update***, ***delete***) ens poden ajudar en aquesta tasca.

**b) Dividir històries complexes**

Això és menys obvi. Són les històries que són molt complexes tècnicament o que no s'han fet mai i no sabem quant ens pot costar. La divisió recomanada en aquest cas és **investigar**, **realitzar** i **refactoritzar**. Podem fer una història d'usuari que sigui "investigar com es fa un registre d'usuari en la nova tecnologia" i s'assignaria un *time-box* limitat, una altra "realitzar la funcionalitat" i encara una altra de " refactoritzar[1] la funcionalitat".

(1)El terme *refactoritzar* significa revisar el codi programat per a poder-lo fer més ràpid, mantenible a llarg termini i entenedor per altres programadors, etc.

**Història d'usuari d'"investigar"**

És important que si es decideix definir una història d'usuari d'"investigar" tingui una finalitat. Pot ser un document amb conclusions, una versió "tosca" de la funcionalitat, però amb els detalls primordials que s'han investigat. Si no indiquem una finalitat, aquesta història pot comportar una pèrdua de temps.

És possible que ens trobem amb històries massa petites, com arreglar algun *bug* o corregir literals. Detectarem una història massa petita de seguida, perquè l'equip de producció dirà que és més ràpid fer-la que no pas escriure-la.

Hi ha algunes històries que fan referència a tot el projecte; per exemple, "la web ha de ser ràpida". Això no és una història, sinó una ***constraint***, és a dir, una condició que han de complir totes les històries que es desenvolupin i, en general, tot el projecte. "Cap pàgina de la web no ha de trigar més de tres segons a carregar-se" seria un bon exemple de *constraint*.

Les *constraints* són presents en tot el projecte i l'equip de producció les ha de tenir sempre en compte abans de finalitzar una història d'usuari.

**6) Verificable**

Com podem saber si una història està acabada si no es pot provar? Una història ha de tenir unes certes condicions de validació que justifiquin que la història està acabada. S'ha de poder provar que una història funciona.

## Els beneficis de les històries d'usuari

Les històries d'usuari comporten, entre d'altres, els beneficis principals següents:

### 1) Promouen la comunicació verbal

Els éssers humans tenen una gran tradició oral. Contes i llegendes han estat transmesos oralment entre generacions, però en algun moment –al voltant del 1970– es va començar a sentir la necessitat de registrar-ho tot per escrit. Hem canviat el fet de compartir coneixement pel de compartir documents. Sembla que deixar una cosa per escrit eviti desacords i malentesos, i no és així.

**La comunicació verbal**

Si en un restaurant veiem que de primer plat serveixen "bacallà o llom amb guarnició", significa que tant la carn com el peix vénen acompanyats de guarnició? Si ja ens costa llegir el menú en el restaurant, imagineu-vos la definició d'una funcionalitat d'un projecte de producció multimèdia.

La paraula *hauria* no s'ha de fer servir mai. Què vol dir "el sistema hauria d'avisar si l'usuari ha introduït un DNI incorrecte"?, significa que el sistema pot no avisar?

Quan el client i el proveïdor fan servir la comunicació oral es produeix un *feedback* constant, del qual es treu molta més informació que per escrit. Promou l'aprenentatge i l'entesa mutus.

### 2) Són comprensibles per tothom

Com que una història d'usuari està escrita en llenguatge natural, la poden comprendre tant els usuaris com l'equip de producció i tothom que participi en el projecte. Promouen la participació i l'opinió de tota classe de perfils, tant dissenyadors com programadors, comercials, etc. No es restringeix l'opinió a cap persona que llegeixi la funcionalitat.

### 3) Tenen la mida justa per a planificar

Ni massa gran ni massa petita, una història d'usuari té la mida justa perquè tant el client com els desenvolupadors s'hi sentin còmodes, tant per a definir-la com per a desenvolupar-la, provar-la o demostrar-la al client.

### 4) Funcionen molt bé en el procés de producció iteratiu

En la gestió àgil de projectes, en què es treballa d'una manera empírica, iterativa i incremental, les històries d'usuari són la millor unitat mínima de funcionalitat per a desenvolupar. Iteració rere iteració, es poden fer servir per a construir el projecte multimèdia final.

### 4.2. Planificació de l'*sprint*

Al començament de cada *sprint* es porta a terme la **reunió de planificació de l'*sprint***. En la primera part de planificació, el propietari del producte i l'equip (amb la facilitació de l'Scrum màster) revisen els elements d'alta prioritat en la pila del producte que el propietari del producte està interessat que es facin en aquest *sprint*. Es discuteixen els objectius i el context d'aquests elements d'alta prioritat de la pila de producte, i es dóna a l'equip informació sobre la idea que té el propietari del producte sobre cada element.

El propietari del producte i l'equip també fixen la definició de fet (*definition of done* o DoD en el terme original en anglès).

Aquesta **definició de fet** és un document breu en el qual s'indiquen les condicions generals que totes les funcionalitats han de complir per a ser donada com a "feta".

Aquesta definició de fet garanteix la transparència i la qualitat apta per a la finalitat del producte i l'organització.

**Definició de fet**

Per exemple, "la funcionalitat ha d'estar traduïda en tres idiomes", o bé "la funcionalitat compleix els mínims de seguretat especificats pel departament de sistemes" podríen ser casos de condicions del document de definició de fet.

La **primera part** de la reunió de planificació de l'*sprint* se centra en la comprensió del que vol el propietari del producte. D'acord amb les regles de Scrum, al final de la primera part el propietari del producte (sempre ocupat) pot deixar la reunió tot i que ha d'estar disponible (per exemple, per telèfon) durant la segona part de la reunió. No obstant això, és benvingut i recomanable que assisteixi també a la segona part.

La **segona part** de la reunió de planificació se centra a planificar les tasques detallades segons la manera d'aplicar els elements que l'equip ha decidit assumir. L'equip selecciona els elements de la pila de producte que es comprometen a realitzar per al final de l'*sprint*, començant a la part superior de la pila de producte (és a dir, a partir dels elements que són la màxima prioritat per al propietari del producte) i baixant la llista en ordre.

El propietari del producte no té control sobre la quantitat de feina que l'equip es compromet a dur a terme; tanmateix, ell o ella sap que els elements a què s'ha compromès l'equip s'han extret de la part superior de la pila de producte que són els elements que ell o ella ha qualificat com els més importants.

L'equip té l'autoritat per a seleccionar també els temes de més avall de la llista després de consultar-ho amb el propietari del producte. Això succeeix quan l'equip i el propietari del producte s'adonen que alguna cosa de prioritat més baixa s'adapta fàcilment i d'una manera més adequada a les qüestions d'alta prioritat que es faran en aquest *sprint*.

La reunió de planificació de l'*sprint* es limita a quatre hores per a un *sprint* de quatre setmanes i dues hores per a un *sprint* de dues setmanes. Per això, l'equip ha d'ajudar el propietari del producte mitjançant l'estimació de la mida de les històries abans de la reunió de planificació de l'*sprint*.

Una vegada es determina la capacitat dels equips disponibles, l'equip comença amb el primer element de la pila de producte –dit d'una altra manera, el que el propietari del producte ha fixat de prioritat més elevada– i es descompon en tasques individuals que es registren en la pila de l'*sprint* o *sprint backlog*. Com s'ha esmentat, el propietari del producte ha d'estar disponible durant la segona part (per exemple, per mitjà del telèfon) de manera que els aclariments i les decisions pel que fa als enfocaments alternatius són possibles. Al final de la reunió, l'equip ha elaborat una llista de tasques amb les estimacions (en general, en hores o fraccions d'un dia). La llista és un punt de partida, però poden sortir més tasques que es desconeixien a mesura que l'equip avança durant l'*sprint*.

Scrum anima que els equips de treball siguin **polivalents**, en comptes de només "treballar en el que diu la teva etiqueta", és a dir, un "programador" només programa, o un "provador" només fa proves. En altres paraules, els membres de l'equip segueixen el principi d'"anar on hi ha la feina" i ajudar de la manera que es pugui. Si hi ha moltes tasques de prova, tots els membres de l'equip poden ajudar a provar. Això no implica que tothom és generalista –sens dubte, algunes persones són especialment hàbils en la prova, i així successivament–, sinó que l'equip ha demostrat que és un enfocament valuós per a l'**intercanvi de coneixements**.

Dit tot això, hi ha moments en què només un membre de l'equip pot fer una tasca determinada, ja que es perdria massa temps si ho fes un altre membre, o no és possible que d'altres aprenguin a fer-ho en un temps raonable –potser ell o ella és l'única persona amb una habilitat artística per a dibuixar. En aquest cas –que és possible, però s'ha de mirar d'evitar perquè depenem d'un sol membre de l'equip per fer una feina específica–, pot ser necessari preguntar-se si les tasques planificades de dibuix que ha de realitzar aquest membre de l'equip són factibles dins de l'*sprint* curt.

Un dels pilars de Scrum és que, una vegada l'equip adopta un compromís, les addicions o canvis han de ser ajornats fins al pròxim *sprint*. Això significa que si a la meitat de l'*sprint* el propietari del producte decideix que hi ha un element nou que li agradaria que l'equip treballés, el propietari no pot fer el canvi fins a l'inici de l'*sprint* següent. Si una circumstància externa sembla que

canvia d'una manera considerable les prioritats, i significa que l'equip perdria el temps si continua treballant, el propietari del producte o l'equip pot acabar l'*sprint*. L'equip s'atura i s'inicia una nova reunió de planificació de l'*sprint*. El cost de fer això (el que es diu **trencar un *sprint***) en general és gran, i això serveix com un desincentiu perquè el propietari del producte o l'equip recorrin a aquesta decisió dràstica.

Si segueix aquestes regles de Scrum, el propietari del producte guanya dues coses:

**1)** En primer lloc, té la confiança de saber que l'equip ha adquirit un compromís per a completar un conjunt realista i clar de les tasques que han triat. Amb el temps, un equip pot arribar a ser molt hàbil en l'elecció i el lliurament d'un compromís realista.

**2)** En segon lloc, el propietari del producte fa els canvis que li agrada a la pila de producte abans del començament de l'*sprint* següent. En aquest moment, és possible i acceptable fer addicions, supressions, modificacions i reordenació de preferència de tot això. Per bé que el propietari del producte no té privilegis per a modificar els elements que han estat seleccionats en l'*sprint* actual, sí que pot fer canvis en les funcionalitats que encara no han entrat a l'*sprint*. És a dir, el propietari del producte ha d'estar sempre un *sprint* de distància per davant de l'equip de desenvolupament per a fer qualsevol modificació. D'aquesta manera s'acaba amb l'estigma al voltant del canvi –canvi de direcció, canvi de requisits o, simplement, canvi d'opinió.

### 4.3. El *daily stand-up*

Una vegada ha començat l'*sprint*, l'equip s'implica en una altra de les claus de les pràctiques de Scrum: el ***daily stand-up***. Es tracta d'una reunió breu (quinze minuts, aproximadament) que es duu a terme cada dia de feina al mateix lloc i a la mateixa hora. Hi assisteixen tots els membres de l'equip. S'anomena així perquè, com que és breu, es recomana que tothom romangui dret. És l'oportunitat de l'equip per a informar els altres de l'avenç assolit i dels obstacles que s'han trobat. En el ***daily* Scrum**, un per un, cada membre de l'equip informa, únicament, dels tres aspectes següents als altres membres de l'equip:

**1)** El que han aconseguit fer des de l'última reunió.

**2)** El que estant fent i que està planificat que acabin abans de la propera reunió i si hi ha algun canvi en l'estimació inicial per a completar la tasca.

**3)** Els obstacles que es troben en el camí.

Tingueu en compte que el *daily* Scrum no és una reunió per a informar un gestor de l'avenç, sinó un temps perquè un equip autoorganitzat comparteixi amb els altres el que passa, per ajudar a coordinar la feina entre tots i optimitzar la probabilitat de complir els compromisos.

Es pren nota dels impediments i l'Scrum màster és responsable d'ajudar els membres de l'equip a resoldre'ls. No hi ha discussió en el *daily* Scrum, només la presentació d'informes en forma de respostes a les tres preguntes. Si és necessari discutir sobre un punt, la discussió es porta a terme immediatament després del *daily* Scrum en una reunió de seguiment, en general només amb les persones que hi estan directament implicades. Aquesta reunió de seguiment és un esdeveniment comú en què l'equip s'adapta a la informació que ha escoltat en el *daily* Scrum; es tracta de la inspecció i l'adaptació del cicle.

En general, es recomana que els gerents o altres persones en posicions d'autoritat no assisteixin al *daily* Scrum o, si ho fan, que no puguin intervenir. Hi ha el risc que l'equip se senti "controlat" –sota la pressió de reportar avenços importants cada dia, la qual cosa és una expectativa poc realista, i s'inhibeix sobre l'informe de problemes–, tendeix a soscavar l'autogestió de l'equip i convida a la microgestió.

### 4.4. Actualització de la pila de l'*sprint* i la gràfica de *sprintburn-down*

Com hem comentat anteriorment, cada dia els membres de l'equip es reuneixen i, entre d'altres, actualitzen l'estimació de la quantitat de temps que resta per a completar la seva tasca actual en la **pila de l'*sprint***. Després d'aquesta actualització, se sumen les hores que queden per a l'equip en conjunt i es representa en la **gràfica de *sprint burn-down***.

Aquesta gràfica mostra, cada dia, una nova estimació de la quantitat de feina (mesurada en hores de la persona o segons els punts relatius) i es manté fins que les tasques de l'equip s'hauran acabat. Idealment, aquesta gràfica és una corba descendent que es troba en una trajectòria fins a arribar a "zero esforç pendent" l'últim dia de l'*sprint*. Per això s'anomena **diagrama de crema** (de tasques). I, encara que de vegades es veu bé, sovint no és així; aquesta és la realitat del desenvolupament del producte. L'important és que mostra a l'equip el progrés que ha fet cap a la meta que s'ha d'assolir, no en termes de la quantitat de temps emprat en el passat (un fet irrellevant en termes de progrés), sinó en termes de la quantitat de treball que resta per fer en el futur –el que separa l'equip del seu objectiu. Si la línia no es "crema", és a dir, la gràfica de "feina que s'ha de fer" respecte del "temps que falta" no té un pendent negatiu (si no decreix cap a la finalització a la meitat de l'*sprint*), l'equip pot necessitar executar algun procediment per a corregir la situació:

**1)** Canviar l'enfocament de la feina o eliminar els obstacles per augmentar la velocitat.

**2)** Buscar ajuda d'algú de fora de l'equip.

**3)** Reduir l'abast del treball.

**4)** Avortar l'*sprint* si es considera que no és possible continuar avançant.

### 4.5. El refinament del producte cartera

Una de les activitats menys conegudes, però mes valuoses en Scrum, és que a cada *sprint* el propietari del producte i l'equip haurien de dedicar una petita part de l'esforç al refinament (o *grooming*) de la pila de producte. Això inclou:

**1)** L'anàlisi detallada dels requisits.

**2)** La divisió d'elements de grans dimensions en altres de més petits.

**3)** L'estimació dels elements nous.

**4)** La reestimació dels elements existents.

Una reunió regular setmanal amb el propietari del producte és suficient. Aquesta activitat no és el refinament dels elements seleccionats per a l'*sprint* actual, sinó que és per als elements del futur, molt probablement en els pròxims un o dos *sprints*. Amb aquesta pràctica, la planificació de *sprint* esdevé relativament simple, perquè el propietari del producte i l'equip Scrum inicien la planificació amb un conjunt d'elements clar, ben analitzat i calculat.

Un signe que aquest procés de refinament no s'està fent (o no s'està fent bé) és que la planificació de l'*sprint* inclogui preguntes importants, el descobriment de noves històries o la confusió sobre les existents.

### 4.6. Finalització de l'*sprint*

Un dels principis bàsics de Scrum és que la durada de l'*sprint* no s'estén més enllà de la data assignada, independentment que l'equip hagi completat la feina a la qual s'ha compromès. Els equips normalment es comprometen a fer més del que poden realitzar en els primers *sprints* i no compleixen els objectius. No obstant això, en el tercer o quart *sprints*, l'equip normalment s'ha adonat del que és capaç de lliurar i compleix els objectius de l'*sprint*.

S'anima els equips a prendre un període de temps per als *sprints* (per exemple, dues setmanes) i que no ho canviïn. Una durada consistent ajuda l'equip a aprendre que pot aconseguir moltes coses, fet que contribueix considerablement a l'estimació i la planificació a llarg termini. També ajuda l'equip a assolir un ritme de feina; el que seria el "batec" de l'equip en Scrum.

### 4.7. Revisió de l'*sprint*

Un cop finalitzat l'*sprint*, es fa la **revisió de l'*sprint***, en què l'equip el revisa amb el propietari del producte. Això sovint és mal etiquetat com la ***demo***, però aquest terme no reflecteix la veritable intenció d'aquesta reunió.

> Una idea clau en Scrum és inspeccionar i adaptar, per veure i saber el que succeeix i després evolucionar a partir de la informació, en la repetició de cicles.

La revisió de l'*sprint* és una activitat d'inspecció i adaptació del producte. És un temps perquè el propietari del producte i les principals parts interessades aprenguin el que succeeix amb el producte i amb l'equip (és a dir, una revisió de l'*sprint*), i perquè l'equip sàpiga què succeeix amb el producte i el mercat. En conseqüència, l'element més important de la revisió és una conversa amb profunditat i la col·laboració entre l'equip i el propietari del producte per a conèixer la situació, per a obtenir consells, entre d'altres. La revisió inclou una demostració del que l'equip ha construït durant l'*sprint*, però si l'enfocament de la revisió és una demostració més que una conversa, hi ha un desequilibri.

En aquesta reunió, hi són presents el propietari del producte, els membres de l'equip i l'Scrum màster, a més dels clients, les parts interessades, experts, executius i qualsevol altra persona interessada.

Una guia de Scrum és que s'ha de gastar tan poc temps com sigui possible en la preparació per a la revisió de l'*sprint*; Scrum suggereix dues hores, com a màxim. És simplement una demostració del que s'ha construït. Durant la revisió, tothom és lliure de fer preguntes i dir-hi la seva.

### 4.8. Retrospectiva de l'*sprint*

La revisió de l'*sprint* implica inspeccionar i adaptar en relació amb el producte. La **retrospectiva de l'*sprint***, que segueix a la revisió, implica revisar i adaptar en relació amb el procés.

Alguns equips se salten aquesta pràctica, la qual cosa és inacceptable perquè l'autoorganització requereix la reflexió freqüent ordinària prevista per la retrospectiva. És el mecanisme principal per a la visibilitat que Scrum proporci-

ona en les àrees de possible millora, a fi de convertir-ho en resultats. És una oportunitat per a l'equip Scrum de discutir el que funciona i el que no funciona, i posar-se d'acord en els canvis que cal intentar.

Scrum màster pot actuar com un facilitador eficaç per a la retrospectiva, però potser és millor trobar un foraster neutral per a facilitar la reunió.

Un bon enfocament és que els Scrum màsters facilitin retrospectives dels altres, cosa que permet la "pol·linització" encreuada entre els equips.

**La retrospectiva de l'*sprint***

Una manera senzilla d'estructurar la retrospectiva de l'*sprint* és escriure quatre columnes en una pissarra, amb les etiquetes següents:

**1)** Què ha sortit bé?

**2)** Què podria haver estat millor?

**3)** Coses per provar?

**4)** Problemes per escalar?

I, després, anar voltant per la sala i que cada persona afegeixi un o més elements a les llistes: idees i experiències, positives i negatives, corresponents a cada una de les columnes. Atès que molts conceptes es repetiran, es poden afegir marques a les repetides, de manera que els elements comuns queden clars.

Després, l'equip busca les causes subjacents i s'arriba a unes conclusions i unes accions de millora, d'acord amb un petit nombre de canvis per tractar en l'*sprint* següent, juntament amb el compromís de revisar els resultats en la propera retrospectiva de l'*sprint*.

Una pràctica útil al final de la retrospectiva és que l'equip identifiqui cadascun dels elements de cada columna amb les etiquetes següents:

- **S**, si és causada per Scrum (és a dir, sense Scrum no passaria).
- **E** a estar exposat per Scrum (és a dir, succeiria amb pila de producte o sense, però Scrum el dóna a conèixer a l'equip).
- **N**, si no està relacionat amb Scrum (com el clima).

Habitualment, l'equip pot anotar una gran quantitat de **S** en "el que funciona bé" al costat del tauler, i una gran quantitat de **E** en "el que podria funcionar millor", la qual cosa és una bona notícia. Si "el que podria funcionar millor" és la llista més llarga, es tracta d'un bon resultat, ja que el primer pas per a resoldre els problemes subjacents és fer-los visibles, i Scrum aconsegueix fer sortir a la llum els problemes ocults en l'organització.

## 4.9. Actualització de la pila de producte i *product burn-down*

En aquest punt, alguns elements s'han acabat, d'altres s'han afegit i encara uns altres tenen noves estimacions, i alguns han caigut de la finalitat fixada. El propietari del producte és responsable d'assegurar que aquests canvis es reflecteixen en la **pila de producte**.

A més a més, Scrum inclou una gràfica, anomenat *product burn-down*, que mostra el progrés cap a la data de llançament. És anàloga a la gràfica de *sprint burn-down*, però és en el nivell més alt dels elements (històries) en comptes de ser

en les tasques de l'*sprint*. Com a propietari de producte novell, és poc probable que sàpiga per què o com ha de crear aquesta taula; és una altra oportunitat per a un Scrum màster d'ajudar el propietari del producte.

### 4.10. *Sprint* següent

Després de la revisió de l'*sprint*, el propietari del producte pot actualitzar la pila de producte amb qualsevol nova idea. En aquest punt, el propietari del producte i l'equip estan a punt per a començar un altre **cicle de *sprint***. No hi ha temps d'inactivitat entre *sprints*.

Els equips normalment van a partir d'una retrospectiva de *sprint*, una tarda, a la pròxima reunió de planificació de *sprint* l'endemà al matí (o després del cap de setmana, per exemple).

Un dels principis del desenvolupament àgil és **ritme sostenible**, i els equips només per les hores de feina regulars en un nivell raonable poden seguir aquest cicle indefinidament.

# Bibliografia

**Cohn, Make** (2009). *Succeeding with Agile. Software development using Scrum* [en línia]. http://www.amazon.com/Succeeding-Agile-Software-Development-Using/dp/0321579364

**The Scrum guide. The official rulebook [en línia].** (). [en línia]. http://www.scrum.org/scrumguides/

**Martin, Robert C.** (2012). *Agile software development: principles, patterns, and practices*. Upper Saddle River, NJ: Pearson Prentice Hall.

**Ruby, Sam** (2009). *Agile web development with rails*. Raleigh, NC: Pragmatic Bookshelf.

**Shore, James** (2008). *The Art of agile development*. Beijing / Sebastopol, CA: O'Reilly Media, Inc.

**Anderson, David James** (2004). *Agile management for software engineering: applying the theory of constraints for business results*. Upper Saddle River, NJ: Prentice Hall PTR, cop. 2004.

**Highsmith, James A.** (2004). *Agile project management: creating innovative products*. Addison-Wesley, cop. 2004.

**Schwaber, Ken** (2004). *Agile project management with Scrum*. Redmond, Wash.: Microsoft Press.

**Schwaber, Ken** (2002). *Agile software development with Scrum*. Upper Seadle River, NJ: Pearson Prentice Hall.