Resiliència a nivell d’aplicació

Principis de Resiliència

La principal font de resiliència fins ara n’és, sense dubte, l’experiència. Aquesta és inherent i varia en funció de cada persona implicada en el desenvolupament del software. Un dels beneficis que dona l’experiència és la resiliència. Com ja hem mencionat, entenem per resiliència la capacitat del software a respondre en circumstàncies adverses i/o sortir d’elles de manera autònoma.

De la resiliència en l’àmbit del software no fa tant 2007[[1]](#endnote-1) que se’n parla, gairebé una dècada. En el seu llibre, *Release it*, Michael T. Nygard tracta el tema de la resiliència software des de la seva experiència. Cal dir que el terme de resiliència ha anat evolucionant. Hi ha tres termes que estan molt relacionats. El primer és el concepte de *Fault Tolerance*, que consisteix en construir software robust. Es a dir, la fallida d’un component que no comporti la caiguda de tot el sistema. El segon concepte és el de la resiliència, tal com s’ha mencionat, és la capacitat recuperar-se. Finalment el concepte d’*Anti Fragility* que és la capacitat d’un sistema no només de tornar a l’estat normal sinó avançar cap a un estat millor.

Nosaltres entenem per aplicació resilient aquella que segueix donant servei a un determinat nivell de qualitat després d’haver patit errors i és capaç de tornar a l’estat normal recuperant la totalitat del servei. En el cas ideal aquesta recuperació és transparent de cara a l’usuari. En els altres casos s’informa l’usuari que la funcionalitat no esta disponible temporalment i es torna a informar quan s’hagi efectuat la recuperació.

# Release it

Ara analitzarem alguns dels patrons que Michael T. Nygard proposa en el seu llibre per aconseguir aplicacions resilient o més resilient. Alguns d’aquests patrons ja venen incorporats als *bons-costums* i/o a *frameworks* corresponents. Des del principi deixa clar que la principal motivació en construir software resilient, finalment arriba a ser la econòmica. El subtítol de la portada ho indica: “*Design and Deploy Production-Ready Doftware.* Segons sosté, una decisió de disseny és una decisió econòmica; i qualsevol *estalvi* que es vulgui fer en aquesta fase tindrà repercussions *cares* en producció. Per tant, la programació ha de ser pragmàtica, orientada a l’entorn de producció, no a l’entorn de proves o QA.

Encara que és difícil trobar-se dos vegades amb el mateix problema, tard o d’hora surten els anti-patrons. Són aquelles situacions sistemàtiques que porten a errors, i per tant es poden aplicar solucions generals. Enfocat en quatre grans temes el llibre estudia en profunditat els problemes que provoquen els anti patrons i els patrons corresponents a les solucions. Els primers dos grans temes compactes són l’estabilitat i la capacitat. Els darrers dos són principis generals de disseny i operacions. Encara i estant enfocat només en l’entorn de producció el llibre aconsegueix donar una visió prou completa del patrons que es podrien aplicar per aconseguir software resilient.

## Estabilitat

Com ja havíem dit el primer tema que tracta és l’estabilitat. El software resilient ha de ser estable. Un error d’una certa funcionalitat no pot ser que ens faci caure tot el sistema, deixant-nos sense poder fer res més abans de reiniciar l’aplicació o el servidor. L’autor identifica una gran varietat d’elements com a anti patrons en aquest tema. Aquests són: els punts d’integració, les reaccions en cadena, cascades d’errors, els usuaris, *threads* bloquejats, atacs d’auto denegació de servei, efectes d’escalat, capacitats no balancejades, respostes lents, SLA’s, respostes no determinades.

Els punts d’integració es van multiplicant conforme el sistema d’informació d’una organització va creixent. Cada cop hi ha més fonts i consumidors d’informació que es necessiten integrar, necessiten interaccionar. Per exemple, CRM, ERP, MRP, BPO entre d’altres. Per tant cada socket, procés, pipe o crida remota pot i arribarà a penjar-se.

Les reaccions en cadena tenen que veure amb temes d’escalabilitat a nivell horitzontal. La figura 3 mostra una granja amb vuit servidors darrere un balancejador de carrega.



Figura 3. Escalabilitat horitzontal.

El problema apareix en cas de caiguda d’un servidor, els que queden s’han de repartir entre tots la seva feina. Depenent del tipus, l’error podria provocar la caiguda d’una altre servidor, fins arribar a caure tot el sistema.

Com a resum esquemàtic dels principis de resiliència que proposa el llibre tenim la figura 2. Aquesta mostra les interaccions de patrons i anti-patrons. Els quadrats representen els patrons i els ovals els anti-patrons.



Figura 2. Interacció entre patrons i antipatrons

El llibre esta enfocat en trobar els anti-patrons i proposar patrons per solucionar la varietat d’errors que provoquen els primers.

# Patterns of resilience

Uwe Frederichson té una sèrie de presentacions sobre la resiliència. Encara que son més recents, 2015, recupera principis que Michael Nygard explica. També menciona i recomana el llibre que encara es considera com la bíblia de la resiliència.

# Resilience is by design

# Principis proposats

Els principis que el present treball ha implementat estan explicats a continuació. (if necessary...)

I will probably mention akka,

1. Considerem el llibre Release it! com el primer en el que no només es parla sino que es tracta la resiliència software d’una manera clara i contundent. [↑](#endnote-ref-1)