Resiliencia a nivell d’aplicació



1. Resiliència Software: Estat de l’art
2. Principis de Resiliència
3. Pla del projecte

# Aplicació

## Servidor: MobService

Per la implementació dels principis de resiliència hem escollit aprofitar una aplicació que és el resultat de l’assignatura de PES anomenada Hangaround. L’aplicació té coma objectiu facilitar l’accés a vida social de les persones amb discapacitats motrius. El cas emblemàtic d’aquest públic objectiu són les persones que utilitzen cadires de rodes. Amb l’ajuda de la comunitat d’usuaris que la fan servir, l’aplicació proporciona informació sobre el nivell d’adaptació d’espais o locals d’accés públic. Segueix el paradigma de client-servidor.

La part servidor és un servei web amb arquitectura API REST, implementada en Java, corre sobre un Tomcat i esta allotjada en Heroku. S’encarrega principalment de la persistència de les dades. Té com a dependència principal l’API de Foursquare que utilitza per a proveir llocs al voltant de la ubicació o direcció cercada. En els casos d’ús només farem servir cerques de ciutats. Foursquare proporciona els llocs: locals, bars, museus, etc. Aquests llocs es guarden en una base de dades relacional en el servidor.

Els usuaris poden fer valoracions en relació amb el nivell d’adaptabilitat dels llocs. Hi ha quatre nivells: UNKNOWN – desconegut, UNADAPTED – sense adaptar, PARTIAL – parcial, i TOTAL. La part servidor determina el nivell d’adaptabilitat d’un lloc en funció de les valoracions que han fet els usuaris sobre aquell lloc. Una valoració requereix tres paràmetres: accés, wc i ascensor.

El primer es refereix a l’accés i mobilitat dins del perímetre del lloc, principalment rampa a l’entrada i amplitud dels passadissos. El segon comprova l’existència de serveis adaptats. Tant el primer com el segon tenen una codificació binaria de **cert** o **fals**. Finalment la presència o absència de l’ascensor és el tercer paràmetre. Aquest només tindria sentit avaluar-lo en el cas en què el local té més d’una planta. La codificació d’aquest paràmetre consisteix en un enumerable amb els següents valors: HAS – el lloc disposa d’ascensor destinat a l’ús públic, NO\_NEED – el lloc només té una planta, finalment HAS\_NOT – el lloc té més d’una planta però només disposa d’escales per accedir.

## Client: Hangaround

En la part client disposem d’una aplicació mòbil per a dispositius android. Esta implementada en Java, aplicació nativa. Com a principals dependències té l’API de Facebook per facilitar l’accés dels usuaris sense la necessitat de crear un compte nou. Els llocs que proporciona Foursquare es representen en un mapa. Per mostrar els llocs s’utilitza l’API de Google Maps.

C:\Users\lucmihai\Documents\TFG-FIB\Photos\resilient-hangaround diagram.png

Figura 2. Nou diagrama de classes: paral·lelisme.

# Casos d’ús

Els principis de resiliència que hem implementat són els següents, mode offline, recuperació d’excepcions NullPointerException i vigilància a nivell de mètode mitjançant una xara neuronal.

La planificació, estimació de costos, desviacions i mesures, avaluació riscos, costs i probabilitats , l’impacte socio-economico-ambiental

1. Execució del projecte

La implementació: descripció de l’aplicació que hem construït, funcionalitats bàsiques, i les implicacions, els problemes (referencies creuades als problemes vists en l’apartat 2) que apliquen i les solucions implementades, nivells de resiliència, cas ideal, cas implementat, contractes, desacoblament i nivell d’intrusió(intrusisme).

Acabem amb conclusions pràctiques de la implementació dels principis. La composició dels tests, automatitzats. Analitzar resultats, comportament de l’aplicació amb la capa de resiliència i sense.

Explicació detallada dels casos d’us

1. Conclusions

Conclusions de la planificació, teòrica dels principis que considerem com a proveïdors de resiliència en el software descripció de l’aplicació que hem construït, autocrítica,

Treball futur i possibles ampliacions.

Disseny en micro artefactes – micro funcionalitats, determina el disseny – arq de microserveis, en lloc de contenidors grans.

I.A – regressió de series números, polinomi de dos graus, a partir de 100 – Deeplearning4j, la xarxa pren el control, capa de resiliència – iOT > corregir sensors,

Meditacions sobre resiliència: mentre estic implementant

Estil escèptic - fred, independent - sense justificacions i amb seguretat!

Temes: nivells

La solució ideal depèn dels casos en els que ens trobem. Considerem dos grans casos. El software ja esta construït o el procés de la seva construcció no ha començat. En el segon cas es pot donar una millor i més amplia solució. Tant si s’aplica la resiliència a nivell intrusiu com si només s’implementa el contracte necessari per la posterior aplicació d’una capa externa i desacoblada de resiliència.

En el segon cas és pot aplicar un cert nivell de resiliència mitjançant solucions externes, però, sempre que l’aplicació compleixi les condicions requerides pels diferents elements: *dokeos* que aporten resiliència a un programari.

El present treball té l’ambició de suggerir principis aplicables en els casos esmentats o en qualsevol altre estat entremig. Només caldrà fer l’anàlisi dels contractes necessaris o dels canvis necessaris.

Encara que no pretenem fixar l’aplicació dels principis de resiliència, en metodologies àgils la resiliència podria generar un backlog propi o simplement afegir històries d’usuari per la creació dokeos.

Resiliència, es pot encarregar de gestionar i fer complir les pre condicions de les funcionalitats d’un software.

Cal tenir en compte que el conjunt de dokeos implementats aquí són caracteristics per a aplicacions mòbils. La gran majoria dels principis serveixen per qualsevol programari però els dokeos implementats no estan desacoblats d’Android.

Paraules clau:

dokeo - la implementació d’un element que prové d’un principi teòric de resiliència.

dokimos - conjunt de dokeos que proporcionen la resiliència d’una aplicació.