POLITECNICO DI TORINO

Department of Control and Computer Engineering
Master of Science in Computer Engineering

Master's Degree Thesis

# Seismic vulnerability assessment of buildings on mobile system

Kotlin Android application and Python server for collecting and storing building data

**Supervisors:**
prof. Marco Torchiano
prof. Fabrizio Barpi
dr. Luca Ardito

**Candidate**
Matteo LIERA

December 2017

# Summary

Just a few years ago, mobile technologies revolutionized everyone's lives. The main advantage of these new devices and features is the ability of interacting between entities in a very fast and reliable way. All these provided features and advantages will be used in this thesis project to create a valid mobile asset tool at service of technicians and professional personnel whose task is to assess the seismic vulnerability of buildings on the Italian territory allowing, in this way, a faster, easier and shared assesment process. The tool that will be implemented and described in detail is an Android application used to quickly and accurately collect information during the activity of seismic surveying of a building on the Italian territory performed by civil engineers. The final purpose is to create a national census of the Italian buildings that would allow the predisposed entities to organize targeted interventions in the most critical regions of the territory, possibly avoiding useless and tragic loss of human lives that unfortunately, especially in the last years (2016-2017) are still happening. To accomplish goal will be also implemented a server capable of receiving report assessments made through the Android application, allowing a safe storage of data for academical purpose.

# Acknowledgements

I would like to thank Professor. Fabrizio Barpi and Phd. Luca Ardito from Politecnico di Torino who gave me the opportunity to realize this thesis project even if I was working in another city. A special thanks to Denis Zucchini who helped me complete the project and supported me everyday since end of July 2017 .

A very special thanks to my family and the 3 m's who, with a lot of patience, endured me to this day.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Mobile technologies are growing at extraordinary speed, allowing more and more people to be connected and to exchange information about every sort of things. Their increasingly success between young but not only, is drawing everyone's attention in the software engineering world towards them; in this way more and more apps and features are available to the public, but one question arise. How many of these applications are really making the difference, a true difference to general welfare? Probably the answer is: a few. The purpose of this mobile application is to make a difference, to help people who work in the civil engineering sector to assess the seismic vulnerability of buildings. This affects all of us, every people that lives in a building must be concerned regarding the safeness of its own house.

This project stems thanks to the cooperation between two departments in the Politecnico di Torino: the civil engineer and software engineer departments, becoming a comprehensive multidisciplinary project and showing again how important to cooperate between competent staff in everyday life.

The strictly software engineer part, that will be addressed in this paper, sees its fulfilment in the development of a mobile tool capable of assisting qualified personnel in the proceedings of seismic vulnerability assessment. The requirement for this tool are mainly three:

1. Being fast in collecting valuable data regarding the building and displaying them in a proper way.
2. Being sufficiently accurate in the evaluations.
3. Group all the functionalities of mobile devices like taking pictures and videos, draw sketches or recording audio in one place to make them easily available to the user.

All the requirements above will be furtherly discussed in details in the following pages.

Nevertheless, to achieve the noble purpose of this project and to make it a bit different from the usual Android application, I decided, and Luca Ardito agreed, to develop the application with the brand new, and to me unknown, Kotlin language. This made everything a bit challenging, since it's a language that was first unveiled in 2011 by JetBrains and that only in 2017 Google announced full support on Android for Kotlin developers.

Finally, the complete system for the project contemplates a very simple web server, capable of a rudimental register/login actions and, most important, a reliable and safe way to store all the data that the technicians that will join the census project will provide. A major concern will be privacy, because the seismic evaluation includes information about the position of the subject building and this is a very sensible data to the public.

This thesis project however is not about security concerns but instead it wants to explain in detail the software engineering process of software developing in a multidisciplinary context, when multiple and heterogeneous (in this case two) co-operates. This paper will be split in three different parts, the first part will address the reasons, the methodologies and the tools that has been choose to fulfil it.

The second part will be all about software development, from requirement engineering to the result, including mobile software architecture and unit testing writing.

The third and last contains an analysis of the results, and the comparison against current technologies for seismic building risk evaluation, showing that is possible to use a mobile device to effectively and efficiently evaluate the seismic hazard of a building.

Last but not least I would like to suggest a successive possible step for this thesis project. With the data collected on the server, it would be fairly easy to build and to train a neural network capable to analyse data and to anticipate the evaluation result using just few inputs instead of the whole report. A very interesting activity would be analyse how fast and how effectively the neural network would get a pretty accurate result and if it could actually be a valid allied to assess and verify the seismic hazard on Italian soil.

# Chapter 2

# Seismic hazard in Italy

## 2.1 Seismic hazard definition

In this chapter will be explained what is the seismic hazard and why it is so important to evaluate the seismic vulnerability of buildings, but first it is important to clarify some basic concept about seismic hazard. In particular the word seismicity indicates how often and how powerful earthquakes can strike an area. If this parameter is known, it is possible to evaluate seismic hazard in terms of probability of a high-power earthquake in a certain period of time. More formally seismic hazard is a composition of three parameters: dangerousness, vulnerability, exposure. Italy has a high dangerousness caused by frequency and intensity of earthquakes, combined with a very high exposure, due population density, and a much higher vulnerability since the majority of buildings are obsolete or outdated. Moreover, Italy has an important building heritage, from historical buildings to farmhouses in rural areas that are a proud for the country but that can turn in deadly traps during an earthquake if no recurrent and proper maintenance is performed.

The next map will show in particular which area of the Italian territory has a higher seismic hazard, and as it is clearly visible the centre and south of Italy, especially in the central part, suffer a much higher risk of recurrent and powerful seismic events. Dangerousness can be evaluated mainly using two methods: deterministic and probabilistic. The first is an analysis based on past events in the area and it usually requires a lot of efforts for retrieve reliable data, so the latter is usually preferred. The most used probabilistic method is the Cornell method. It tries to calculate dangerousness by quantifying the degree of seismic activity in predefined sismogenic areas and to calculate the effects from such areas on the territory in relation to the distance from the earthquake epicentre. One last very recent method that deserves to be mentioned is the neo-deterministic NDSHA (Neo-Deterministic Seismic Hazard Analysis).

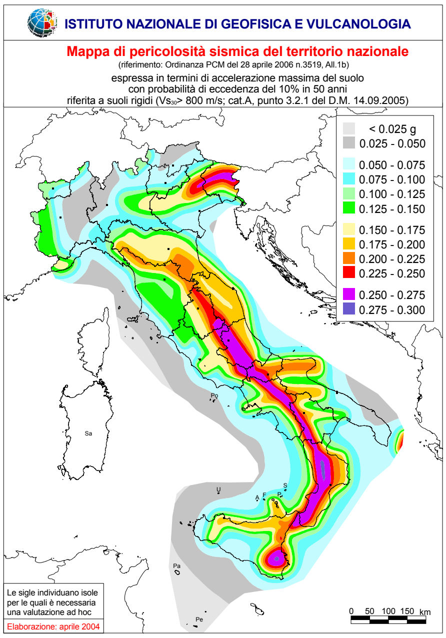



Figure 2.1. Seismic hazard in Italy (in Italian) [1]

The main characteristics of exposure studies are instead population density, goods and infrastructure. Has it's easy to understand, population density is the main parameter and it is very hard to account for. This is blecause it depends from a long list of other factors like time of the day, day of the week, weather and in general all the events or context that may influence people behaviour in the short term, making it very difficult to foretell. Infrastructure incorporate all human made structure and buildings useful to everyday life, communication and civilization. The more strategic infrastructures are concentrated in dangerous area, more the exposure will be high. The last parameter is goods that comprehend all the resources that people needs to survive, like water, electricity and food.

Last parameter needed to define the seismic hazard is vulnerability. It is defined as the probability of a building to get a certain amount of damage during an earthquake. The type and the amount of damage depend on the structural state of the building, considering year of construction, materials, shape of the building itself and even proximity with other buildings. All these factors are in turn depending on the type of earthquake, the type of terrain on which the building is built and how it responds to the shake. The methodologies that try to evaluate vulnerability are expert judgment, statistic or mechanistic. While the first doesn't need explanation

[1]Source: *http : //www.protezionecivile.gov.it/jcms/it/descrizione_sismico*

the statistical methods try to classify buildings using the structural information of the building itself crossed with damages of similar building in past seismic events. The mechanistic methods instead use mathematical models to simulate earthquakes and their effects on existing structures represent by models as well.

It becomes clear that to evaluate seismic hazard in a more efficient and accurate way some tool to collect and store data on seismic events are needed.

## 2.2 The Italian classification

Even today, daily weak earthquakes are hitting the central part of Italy, on the border area between Marche, Umbria and Abruzzo where just a year ago an earthquake of 6.0 magnitude (the equivalent of 63 terajoule, the same energy unleashed on Hiroshima by the atomic bomb); the shakes never stopped and the casualties from August 2016 to January 2017 calculate to the terrible amount of more than three hundred. Even more recently another earthquake hit the island Ischia with a 4.0 magnitude causing 2 casualties, a rather exceptional model of vulnerability and exposure.

Very important becomes how Italian legislation answer to this important hazard to the country. Unfortunately the study of the seismic zone starts right after two tremendous events at the beginning of the twentieth century and contemplated only two categories: municipalities hit by an earthquake and municipalities (still) not hit. Today law are based on the law n.64 dated 2 february 1974 namely *"Provvedimenti per le costruzioni con particolari prescrizioni per le zone sismiche"* that invalidates all previous laws. Only in this year a national law established classification methods for the entire territory and the regulations for the government to publish regularly updates on seismic classification and technical standards. In the next years all the competences regarding the updates of the seismic areas were transferred to regions while the State kept the choice of the standards to use for the various evaluations. Only in 2002 though, with the *"DPR n.380 and 27 dicembre 2002, n.301"* that integrated it, all the buildings must conform to specific standards, according to the seismic zone where they are built.

Italy was then divided in four different seismic zones, ordered by dangerousness and with the obligation that every building must satisfy the standards accordingly to their zone, except for the regions in the less dangerous zone. In this case is the region itself that can choose if order the fulfilment of the standards or not. Important to notice that the division in four zones was requested by the civil protection and not from the minister itself.

The parliament had to mediate between these two entities to align technical standards for private and public infrastructures and procedures for region seismic zone assessments as well. From 2008 to 2012 a sequence of rules were enact consequently to the earthquakes in Abruzzo and Emilia, until 2016 when *Conferenza Unificata* (a union of public administration representatives) decided to prepare a definitive revision of the Technical Standards for Construction (NTC); in this scenario, the need for a single regulatory framework was formally expressed, a regulation to coordinate project and intervention activities for buildings.

In this very year the government introduced a system of tax incentives to promote restoration and improvement interventions behaviour of structures under the action of earthquake, to invite and support people who wanted to make safe their building.

This is just a summary of the laws that aimed to prevent seismic hazard which effectiveness won't be addressed.

## 2.3 Available tools for evaluations

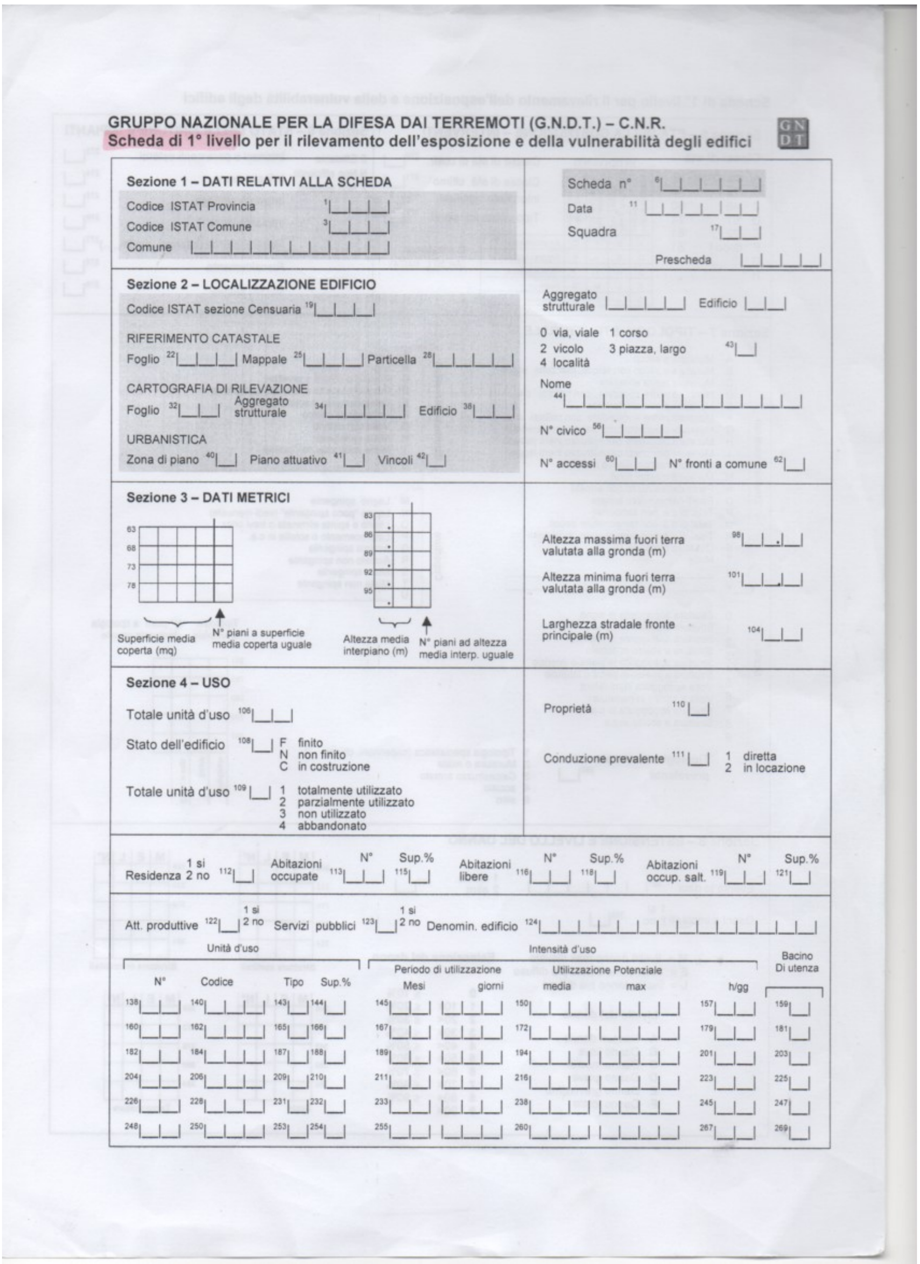
This section will show which are the current methods used by qualified personnel to evaluate the exposition and vulnerability of buildings, officially provided by the GNDT (Gruppo Nazionale per la Difesa dei Terremoti) and CSLP (Consiglio Superiore dei Lavori Pubblici).

### 2.3.1 Evaluation cards

The documents provided by the GNDT are precompiled complex card that can be filled with the data regarding the object building and area. The purpose of these documents is to:

1. Collection of Preliminary Information for Vulnerability Census
2. Vulnerability Census for Masonry or Concrete Buildings;
3. I / II level card for exposure and vulnerability detection of buildings (masonry or reinforced concrete);
4. I / II level card for detection of vulnerability and vulnerability of buildings (sheds, churches, etc.);
5. Level I data sheet gives prompt intervention and ease for ordinary buildings in post-seismic emergency.

The top-level cards are used to urban areas statistical examinations, while the second-level ones are more detailed and used in the examination of a smaller number of buildings. All the relevant cards, in order to facilitate easy cataloguing and thus the creation of an accessible database, have an introductory section where information is provided necessary for the identification of the building (region, province,



GRUPPO NAZIONALE PER LA DIFESA DAI TERREMOTI (G.N.D.T.) – C.N.R.
Scheda di 1° livello per il rilevamento dell'esposizione e della vulnerabilità degli edifici

GN DT

**Sezione 1 – DATI RELATIVI ALLA SCHEDA**

Codice ISTAT Provincia [1]
Codice ISTAT Comune [3]
Comune
Scheda n° [6]
Data [11]
Squadra [17]
Prescheda

**Sezione 2 – LOCALIZZAZIONE EDIFICIO**

Codice ISTAT sezione Censuaria [19]
RIFERIMENTO CATASTALE
Foglio [22] Mappale [25] Particella [28]
CARTOGRAFIA DI RILEVAZIONE
Foglio [32] Aggregato strutturale [34] Edificio [38]
URBANISTICA
Zona di piano [40] Piano attuativo [41] Vincoli [42]
Aggregato strutturale Edificio
0 via, viale 1 corso
2 vicolo 3 piazza, largo [43]
4 località
Nome [44]
N° civico [56]
N° accessi [60] N° fronti a comune [62]

**Sezione 3 – DATI METRICI**

63 68 73 78
Superficie media coperta (mq)
N° piani a superficie media coperta uguale
83 86 89 92 95
Altezza media interpiano (m)
N° piani ad altezza media interp. uguale
Altezza massima fuori terra valutata alla gronda (m) [98]
Altezza minima fuori terra valutata alla gronda (m) [101]
Larghezza stradale fronte principale (m) [104]

**Sezione 4 – USO**

Totale unità d'uso [106]
Stato dell'edificio [108] F finito, N non finito, C in costruzione
Totale unità d'uso [109] 1 totalmente utilizzato, 2 parzialmente utilizzato, 3 non utilizzato, 4 abbandonato
Proprietà [110]
Conduzione prevalente [111] 1 diretta, 2 in locazione

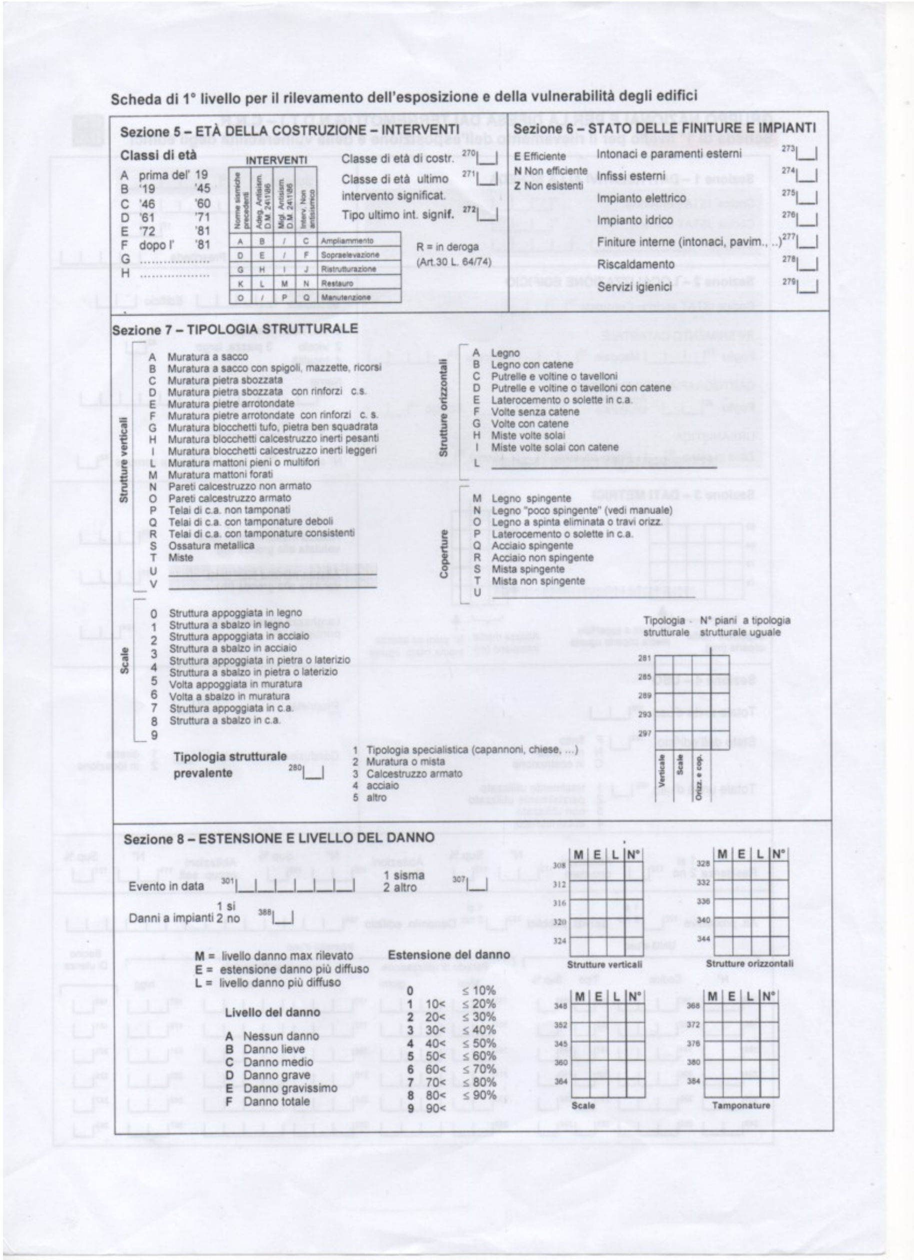
Residenza 1 si 2 no [112] Abitazioni occupate N° [113] Sup.% [115] Abitazioni libere N° [116] Sup.% [118] Abitazioni occup. salt. N° [119] Sup.% [121]

Att. produttive [122] 1 si 2 no Servizi pubblici [123] 1 si 2 no Denomin. edificio [124]

| Unità d'uso: N° | Codice | Tipo | Sup.% | Intensità d'uso – Periodo di utilizzazione: Mesi | giorni | Utilizzazione Potenziale: media | max | h/gg | Bacino Di utenza |
|---|---|---|---|---|---|---|---|---|---|
| 138 | 140 | 143 | 144 | 145 | | 150 | | 157 | 159 |
| 160 | 162 | 165 | 166 | 167 | | 172 | | 179 | 181 |
| 182 | 184 | 187 | 188 | 189 | | 194 | | 201 | 203 |
| 204 | 206 | 209 | 210 | 211 | | 216 | | 223 | 225 |
| 226 | 228 | 231 | 232 | 233 | | 238 | | 245 | 247 |
| 248 | 250 | 253 | 254 | 255 | | 260 | | 267 | 269 |

Figure 2.2. Fist level card, pt.1

city and its ISTAT codes, location on cadastral maps).
The first level card for detecting the exposure and vulnerability of buildings (masonry or reinforced concrete) is generally also considered as a support card for post-earthquake inspection. As a matter of facts, it includes Section 8 dedicated to the extent of the apparent damage extension and damage level while section seven is



Scheda di 1° livello per il rilevamento dell'esposizione e della vulnerabilità degli edifici

**Sezione 5 – ETÀ DELLA COSTRUZIONE – INTERVENTI**

Classi di età

A prima del' 19
B '19 '45
C '46 '60
D '61 '71
E '72 '81
F dopo l' '81
G ..................
H ..................

INTERVENTI

| Norme sismiche precedenti | Adeg. Antisism. D.M. 24/1/86 | Migl. Antisism. D.M. 24/1/86 | Interv. Non antisismico | |
|---|---|---|---|---|
| A | B | / | C | Ampliammento |
| D | E | / | F | Sopraelevazione |
| G | H | I | J | Ristrutturazione |
| K | L | M | N | Restauro |
| O | / | P | Q | Manutenzione |

Classe di età di costr. 270

Classe di età ultimo intervento significat. 271

Tipo ultimo int. signif. 272

R = in deroga (Art.30 L. 64/74)

**Sezione 6 – STATO DELLE FINITURE E IMPIANTI**

E Efficiente
N Non efficiente
Z Non esistenti

Intonaci e paramenti esterni 273
Infissi esterni 274
Impianto elettrico 275
Impianto idrico 276
Finiture interne (intonaci, pavim., ..) 277
Riscaldamento 278
Servizi igienici 279

**Sezione 7 – TIPOLOGIA STRUTTURALE**

Strutture verticali

A Muratura a sacco
B Muratura a sacco con spigoli, mazzette, ricorsi
C Muratura pietra sbozzata
D Muratura pietra sbozzata con rinforzi c.s.
E Muratura pietre arrotondate
F Muratura pietre arrotondate con rinforzi c. s.
G Muratura blocchetti tufo, pietra ben squadrata
H Muratura blocchetti calcestruzzo inerti pesanti
I Muratura blocchetti calcestruzzo inerti leggeri
L Muratura mattoni pieni o multifori
M Muratura mattoni forati
N Pareti calcestruzzo non armato
O Pareti calcestruzzo armato
P Telai di c.a. non tamponati
Q Telai di c.a. con tamponature deboli
R Telai di c.a. con tamponature consistenti
S Ossatura metallica
T Miste
U
V

Strutture orizzontali

A Legno
B Legno con catene
C Putrelle e voltine o tavelloni
D Putrelle e voltine o tavelloni con catene
E Laterocemento o solette in c.a.
F Volte senza catene
G Volte con catene
H Miste volte solai
I Miste volte solai con catene
L

Coperture

M Legno spingente
N Legno "poco spingente" (vedi manuale)
O Legno a spinta eliminata o travi orizz.
P Laterocemento o solette in c.a.
Q Acciaio spingente
R Acciaio non spingente
S Mista spingente
T Mista non spingente
U

Scale

0 Struttura appoggiata in legno
1 Struttura a sbalzo in legno
2 Struttura appoggiata in acciaio
3 Struttura a sbalzo in acciaio
4 Struttura appoggiata in pietra o laterizio
5 Struttura a sbalzo in pietra o laterizio
6 Volta appoggiata in muratura
7 Volta a sbalzo in muratura
8 Struttura appoggiata in c.a.
9 Struttura a sbalzo in c.a.

Tipologia - strutturale; N° piani a tipologia strutturale uguale

| | Verticale | Scale | Orizz. e cop. | |
|---|---|---|---|---|
| 281 | | | | |
| 285 | | | | |
| 289 | | | | |
| 293 | | | | |
| 297 | | | | |

Tipologia strutturale prevalente 280

1 Tipologia specialistica (capannoni, chiese, ...)
2 Muratura o mista
3 Calcestruzzo armato
4 acciaio
5 altro

**Sezione 8 – ESTENSIONE E LIVELLO DEL DANNO**

Evento in data 301

1 sisma
2 altro 307

Danni a impianti 1 si 2 no 388

M = livello danno max rilevato
E = estensione danno più diffuso
L = livello danno più diffuso

Livello del danno

A Nessun danno
B Danno lieve
C Danno medio
D Danno grave
E Danno gravissimo
F Danno totale

Estensione del danno

0 ≤ 10%
1 10< ≤ 20%
2 20< ≤ 30%
3 30< ≤ 40%
4 40< ≤ 50%
5 50< ≤ 60%
6 60< ≤ 70%
7 70< ≤ 80%
8 80< ≤ 90%
9 90<

Strutture verticali

| | M | E | L | N° |
|---|---|---|---|---|
| 308 | | | | |
| 312 | | | | |
| 316 | | | | |
| 320 | | | | |
| 324 | | | | |

Strutture orizzontali

| | M | E | L | N° |
|---|---|---|---|---|
| 328 | | | | |
| 332 | | | | |
| 336 | | | | |
| 340 | | | | |
| 344 | | | | |

Scale

| | M | E | L | N° |
|---|---|---|---|---|
| 348 | | | | |
| 352 | | | | |
| 345 | | | | |
| 360 | | | | |
| 364 | | | | |

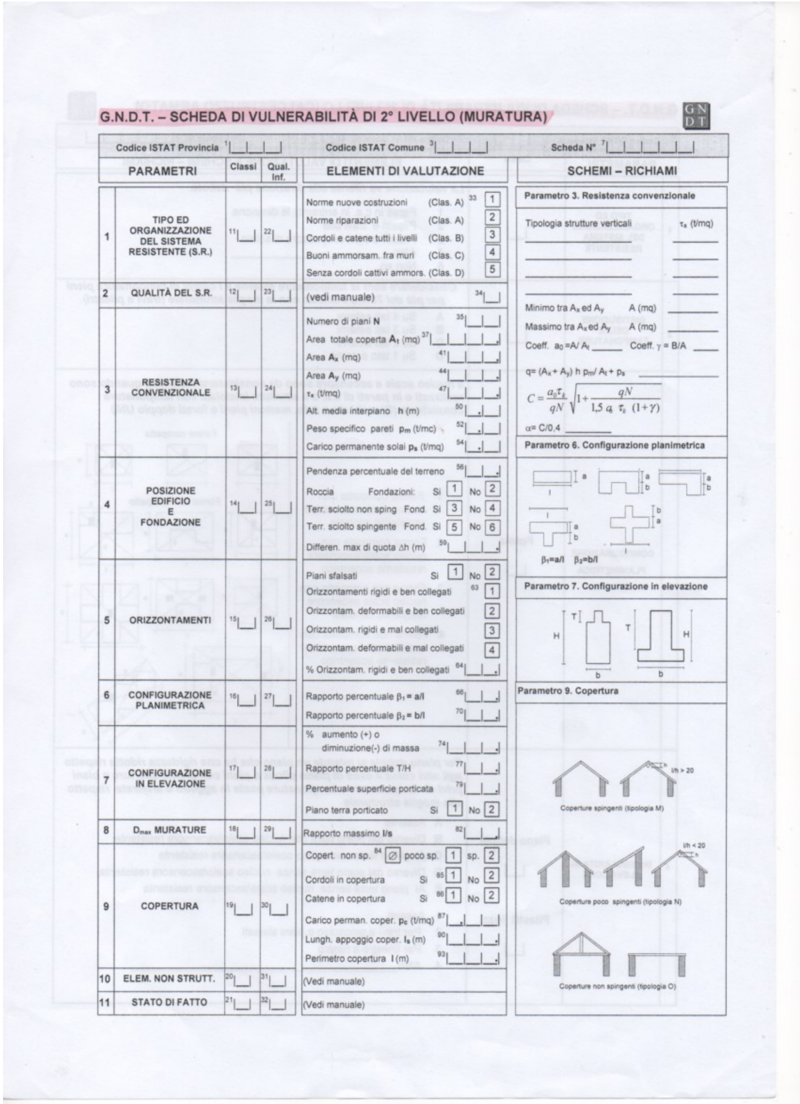
Tamponature

| | M | E | L | N° |
|---|---|---|---|---|
| 368 | | | | |
| 372 | | | | |
| 376 | | | | |
| 380 | | | | |
| 384 | | | | |

Figure 2.3. Fist level card, pt.2

dedicated to the coding of the structural typology. These sections allow to focus on the damage on buildings and the vulnerability factors associated with typology, and therefore allow also to carry out censuses aimed to assess the seismic hazard.

The second level cards are based on a board that collects typological and con-



G.N.D.T. – SCHEDA DI VULNERABILITÀ DI 2° LIVELLO (MURATURA)

GNDT

Codice ISTAT Provincia [1] | Codice ISTAT Comune [3] | Scheda N° [7]

| | PARAMETRI | Classi | Qual. Inf. | ELEMENTI DI VALUTAZIONE |
|---|---|---|---|---|
| 1 | TIPO ED ORGANIZZAZIONE DEL SISTEMA RESISTENTE (S.R.) | 11 | 22 | Norme nuove costruzioni (Clas. A) 33 [1]<br>Norme riparazioni (Clas. A) [2]<br>Cordoli e catene tutti i livelli (Clas. B) [3]<br>Buoni ammorsam. fra muri (Clas. C) [4]<br>Senza cordoli cattivi ammors. (Clas. D) [5] |
| 2 | QUALITÀ DEL S.R. | 12 | 23 | (vedi manuale) 34 |
| 3 | RESISTENZA CONVENZIONALE | 13 | 24 | Numero di piani N 35<br>Area totale coperta $A_t$ (mq) 37<br>Area $A_x$ (mq) 41<br>Area $A_y$ (mq) 44<br>$\tau_x$ (t/mq) 47<br>Alt. media interpiano h (m) 50<br>Peso specifico pareti $p_m$ (t/mc) 52<br>Carico permanente solai $p_s$ (t/mq) 54 |
| 4 | POSIZIONE EDIFICIO E FONDAZIONE | 14 | 25 | Pendenza percentuale del terreno 56<br>Roccia Fondazioni: Si [1] No [2]<br>Terr. sciolto non sping Fond. Si [3] No [4]<br>Terr. sciolto spingente Fond. Si [5] No [6]<br>Differen. max di quota Δh (m) 59 |
| 5 | ORIZZONTAMENTI | 15 | 26 | Piani sfalsati Si [1] No [2]<br>Orizzontamenti rigidi e ben collegati 63 [1]<br>Orizzontam. deformabili e ben collegati [2]<br>Orizzontam. rigidi e mal collegati [3]<br>Orizzontam. deformabili e mal collegati [4]<br>% Orizzontam. rigidi e ben collegati 64 |
| 6 | CONFIGURAZIONE PLANIMETRICA | 16 | 27 | Rapporto percentuale $\beta_1$ = a/l 66<br>Rapporto percentuale $\beta_2$ = b/l 70 |
| 7 | CONFIGURAZIONE IN ELEVAZIONE | 17 | 28 | % aumento (+) o diminuzione(-) di massa 74<br>Rapporto percentuale T/H 77<br>Percentuale superficie porticata 79<br>Piano terra porticato Si [1] No [2] |
| 8 | $D_{max}$ MURATURE | 18 | 29 | Rapporto massimo l/s 82 |
| 9 | COPERTURA | 19 | 30 | Copert. non sp. 84 [∅] poco sp. [1] sp. [2]<br>Cordoli in copertura Si 85 [1] No [2]<br>Catene in copertura Si 86 [1] No [2]<br>Carico perman. coper. $p_c$ (t/mq) 87<br>Lungh. appoggio coper. $l_s$ (m) 90<br>Perimetro copertura l (m) 93 |
| 10 | ELEM. NON STRUTT. | 20 | 31 | (Vedi manuale) |
| 11 | STATO DI FATTO | 21 | 32 | (Vedi manuale) |

SCHEMI – RICHIAMI

Parametro 3. Resistenza convenzionale

Tipologia strutture verticali $\tau_k$ (t/mq)

Minimo tra $A_x$ ed $A_y$ A (mq)

Massimo tra $A_x$ ed $A_y$ A (mq)

Coeff. $a_0$=A/ $A_t$ Coeff. $\gamma$ = B/A

q= $(A_x+A_y)$ h $p_m$/ $A_t$ + $p_s$

$$C=\frac{a_0\tau_k}{qN}\sqrt{1+\frac{qN}{1{,}5\,a_0\,\tau_k\,(1+\gamma)}}$$

α= C/0,4

Parametro 6. Configurazione planimetrica



Parametro 7. Configurazione in elevazione

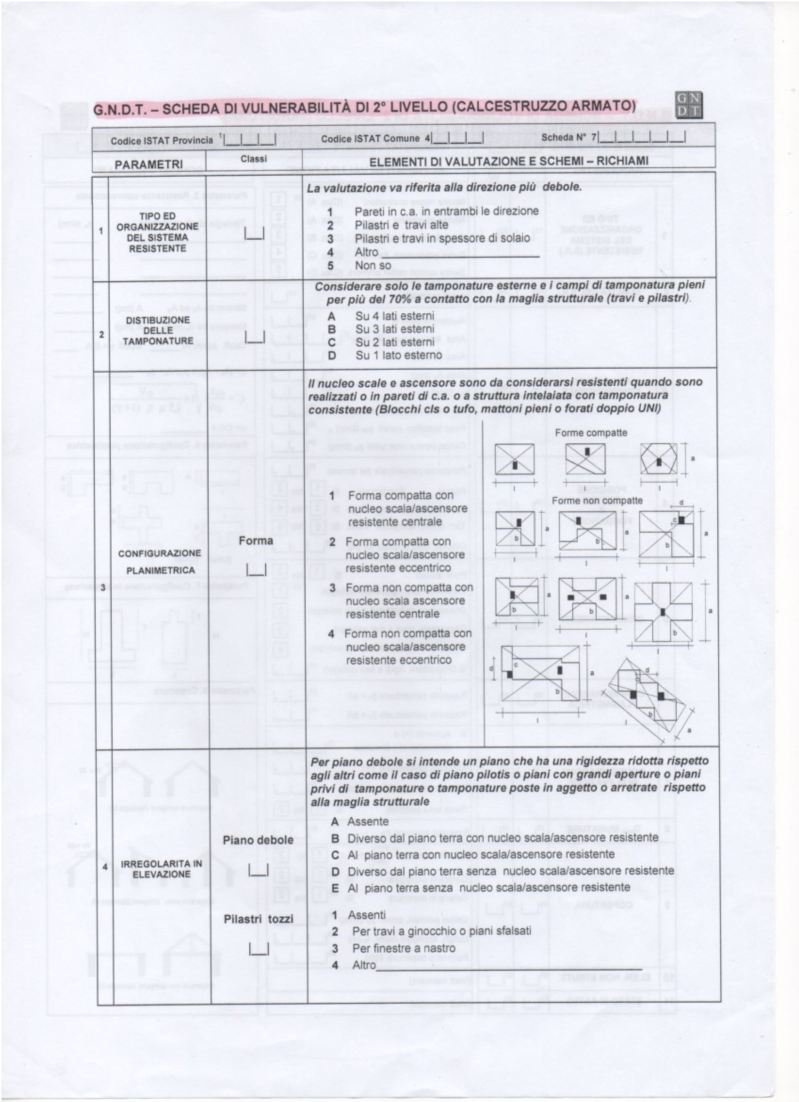


Parametro 9. Copertura



Figure 2.4. Second level card, pt.1

structive information on each single building. Given the accurately relevant information required for compiling this card, the latter is incompatible with emergency management times. In effect, the card is aimed at preventative analysis of seismic vulnerability, for example in support of cost estimates / benefits of building system reinforcement systems. Also, the 2nd level card on one side requires compilation of the first level board, presenting in this way several redundant data, as well as a non-complete use of the information already coded at the first level (see for example information on damage, or those relating to the type of data attic and cover). It is



G.N.D.T. – SCHEDA DI VULNERABILITÀ DI 2° LIVELLO (CALCESTRUZZO ARMATO)

Codice ISTAT Provincia 1|__|__|__| Codice ISTAT Comune 4|__|__|__| Scheda N° 7|__|__|__|__|__|

| | PARAMETRI | Classi | ELEMENTI DI VALUTAZIONE E SCHEMI – RICHIAMI |
|---|---|---|---|
| 1 | TIPO ED ORGANIZZAZIONE DEL SISTEMA RESISTENTE | \|__\| | *La valutazione va riferita alla direzione più debole.*<br>1 Pareti in c.a. in entrambi le direzione<br>2 Pilastri e travi alte<br>3 Pilastri e travi in spessore di solaio<br>4 Altro ______<br>5 Non so |
| 2 | DISTIBUZIONE DELLE TAMPONATURE | \|__\| | *Considerare solo le tamponature esterne e i campi di tamponatura pieni per più del 70% a contatto con la maglia strutturale (travi e pilastri).*<br>A Su 4 lati esterni<br>B Su 3 lati esterni<br>C Su 2 lati esterni<br>D Su 1 lato esterno |
| 3 | CONFIGURAZIONE PLANIMETRICA | Forma \|__\| | *Il nucleo scale e ascensore sono da considerarsi resistenti quando sono realizzati o in pareti di c.a. o a struttura intelaiata con tamponatura consistente (Blocchi cls o tufo, mattoni pieni o forati doppio UNI)*<br>1 Forma compatta con nucleo scala/ascensore resistente centrale<br>2 Forma compatta con nucleo scala/ascensore resistente eccentrico<br>3 Forma non compatta con nucleo scala ascensore resistente centrale<br>4 Forma non compatta con nucleo scala/ascensore resistente eccentrico<br>Forme compatte<br>Forme non compatte |
| 4 | IRREGOLARITA IN ELEVAZIONE | Piano debole \|__\| | *Per piano debole si intende un piano che ha una rigidezza ridotta rispetto agli altri come il caso di piano pilotis o piani con grandi aperture o piani privi di tamponature o tamponature poste in aggetto o arretrate rispetto alla maglia strutturale*<br>A Assente<br>B Diverso dal piano terra con nucleo scala/ascensore resistente<br>C Al piano terra con nucleo scala/ascensore resistente<br>D Diverso dal piano terra senza nucleo scala/ascensore resistente<br>E Al piano terra senza nucleo scala/ascensore resistente |
| | | Pilastri tozzi \|__\| | 1 Assenti<br>2 Per travi a ginocchio o piani sfalsati<br>3 Per finestre a nastro<br>4 Altro ______ |

Figure 2.5. Second level card, pt.2

therefore a procedure not totally autonomous to respect to the 1st level, and inappropriate, despite the complexity of the survey, to provide the necessary information for methodologies based on approximate calculation methods of seismic resistance. The GNDT cards allow for a timely evaluation of the building's building vulnerability, with a score between 0 and 100 (for masonry buildings) and between -25 and 100 (for buildings in c.a). The Vulnerability Index is then computed by adding the contributions of vulnerability scores of 11 parameters detected and related to certain aspects of the seismic behaviour typical of masonry or reinforced concrete constructions. It can therefore be considered a qualitative method.

### 2.3.2 Spettri-NTC

Spettri-NTC [14] is an Excel document, provided by the CSLP useful to calculate the components (vertical and horizontal) of the seismic action on the Italian territory, it is a complex interactive spreadsheet that will be later use as official comparison with the app results. The process to calculate response spectrum in this spreadsheet is divided in three phases. In the first one the user will be asked to insert the specific location of the site to be analysed and it can be done in two ways: using coordinates directly or using the municipality.

In the latter case the interpolation won't be very accurate, since a municipality


Figure 2.6. Spettri-NTC, phase 1: Identifying the site's dangers

could contain several seismic zones. Once the location is set the program will calculate (using an internal database) the default response spectrum for the area. This calculation uses previous data collected with surveys before or right after an earthquake since this is the only way to measure earthquake intensity. Since it is very unlikely to get the precise location of a measurement, an interpolation is needed to be more accurate in the calculation. To implement this interpolation the program chooses the closes surrounding four points forming a square around the selected area. Using the measurement from these four point an estimate is calculated for the selected area.

The second phase allow the user to input information regarding the reference life of the subject building. The nominal life indicates the minimum number of usage year (under ordinary maintenance) of the structure as it was designed: it is specified in the design documents. The use class define the strategic importance of the

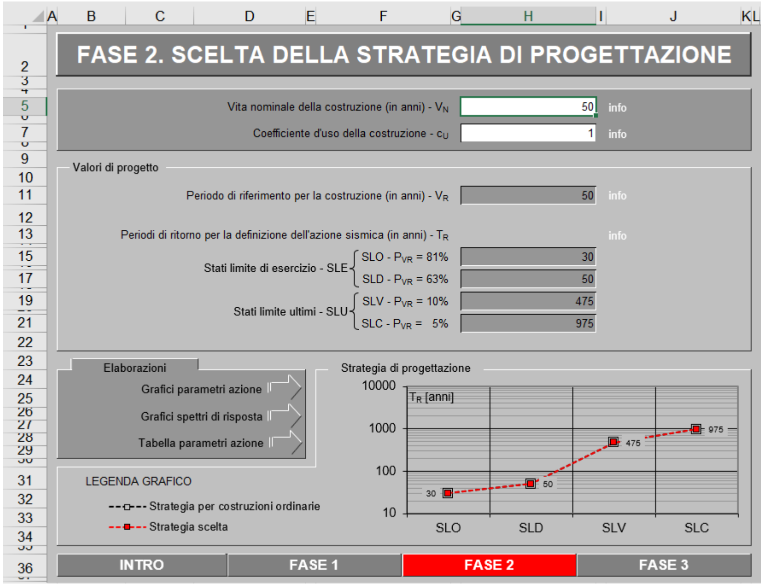



Figure 2.7. Spettri-NTC, phase 2: Choice design strategy

building. These two parameters combined identifies the reference life that will be used to estimate the parameters (again read from the database and interpolated) of the limit states.

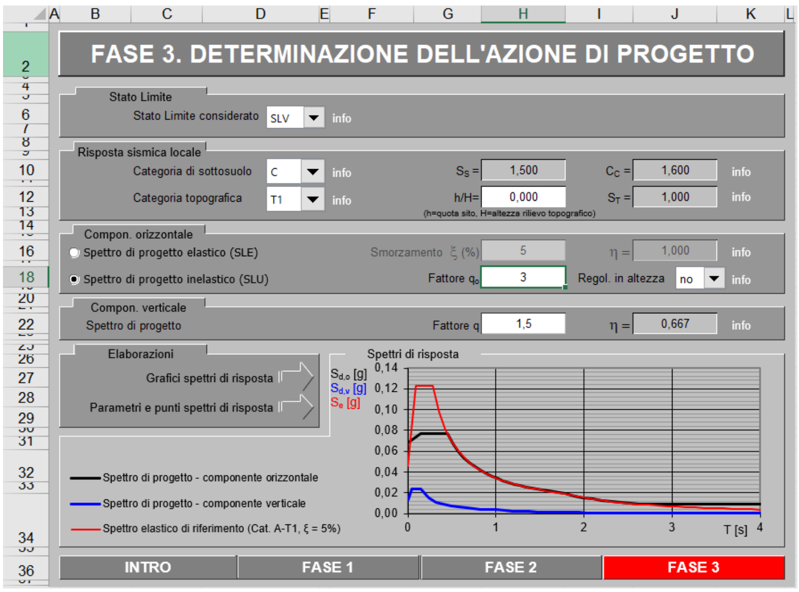



Figure 2.8. Spettri-NTC, phase 3: Project action

The third and last phase is used to specifically calculate spectra of the selected limit state. It allows to select other information about the location like the soil and topographic category, together with the q factor that indicates the structural type, the degree of elasticity and design criteria. This program provides all formulas used in the calculations and these very formulas will be implemented in the application to exactly to reproduce faithfully the official procedure.

## 2.4 The need of a new tool

By analysing today tool it is clear that something more is needed to make the process more lean, efficient and portable because Spettri is just a desktop application, not really suitable for portability and usability in a mobile context. It require lots of input from the final user and it is not automatic, meaning that it cannot automatically acquire the location of the building and calculate the paramters automatically. Moreover Spettri computes just the seismic acceleration for a subject site, it doesn't provide any feature for building analysis. The paper cards instead comes with all the problems associated with physical files: hard to fill, to handle and to share. The solution to overcome all these problems and these weak spots has been the conception of an Android application, a tool in line with modern technologies. The only concern was the difficulty of bringing very complex calculations, interpolation and very technical detail into an Android app that has not been design for such operation and that the high technicity of the app could discourage even the more audacious civil engineer. These concerns were all shut down with the thesis work of Dr. Denis Zucchini, which demonstrate and explains with a very simple prototype how the application should be made, and which calculation are required to reach such target. This thesis work is the natural continuation of Zucchini work, the contribute from Zucchini was basically to be the product manager the application, by indicating how the application should work and by also acting like a beta tester, having the exact same role as the role target of the application.
With the increase in the spread of smartphones and tablets, applications demand has increased considerably. Having effective and innovative tools for companies that need to communicate directly with their organization or target audience is today crucial. The idea of designing an innovative and revolutionary application stems from the need to eliminate the paper sources that are still largely used today, especially in the field of construction, helping to slow the bureaucratic times. The application is designed for use by professionals specially enrolled in Civil Engineering / Architecture. From the technical point of view this application will use a simplified analytical method (Linear Static Analysis) that will quickly return the final vulnerability index.
The achievement of this result will be through a simple, rapid and sequential compilation of data about the intrinsic characteristics of the building and the site on which it is located.
This is possible through assisted data compilation acquired through internal application databases and through a geo-localization system that will quickly compile the seismogenic data of the site and the building itself. It is also provided the possibility to acquire multimedia data by means of picture, video, sketch, note, audio note. A further peculiarity of the application will be, in the future, the ability to operate and interact through data collection and exchange directly with entities such as land

register and other institutions, but can be extended to a wider range of operators who intervene into the process. Once the data is entered, the application provides a final technical report in pdf / docx format with the possibility of transmitting the same to third-party public and / or private users. In a first phase, the application will be designed to work with a subset of buildings, but in a later stage in it could be modified to include more construction technologies and more detailed analysis.

# Chapter 3

# System Development

This chapter will show the whole procedure of software engineering of the system to be, from the requirements collection to the develop of the app and the backend server in support. Remember to read first the section glossary to better understand the terminology and the subjects of the system.

## 3.1 Meeting requirements

In this section will be explained which are the requirements and all the procedure to collect them. It all began with a meeting in the Politecnico where Professor. Barpi and Denis Zucchini explained in a general way how the app should work, and which were the main focuses and concerns about it.
The result of this meeting, in an informal way, was that the system aims at collecting, analysing and displaying useful data for the seismic classification of concrete buildings on Italian territory digitally, through today's mobile technologies. Specifically, using an Android smartphone or tablet app, specialized staff can compile a report of the building in analytical, non-stop and quickly to get a sufficiently reliable result [1], indicating the risk of collapse. Finally, this report can be uploaded to a portal, enabling the census and classification of all buildings in the territory.

Moreover, in this case, the requirements engineering phase has been easier because Denis already had implemented ad demo application that was capable of a part of the operations requested from the system and this was a valid source of requirements. Now let's describe the system in a formal way. First of all, who are

[1]DM 14 January 2008, published on Gazzetta Ufficiale n. 29 del 4 February 2008 - Suppl. Ordinario n. 30

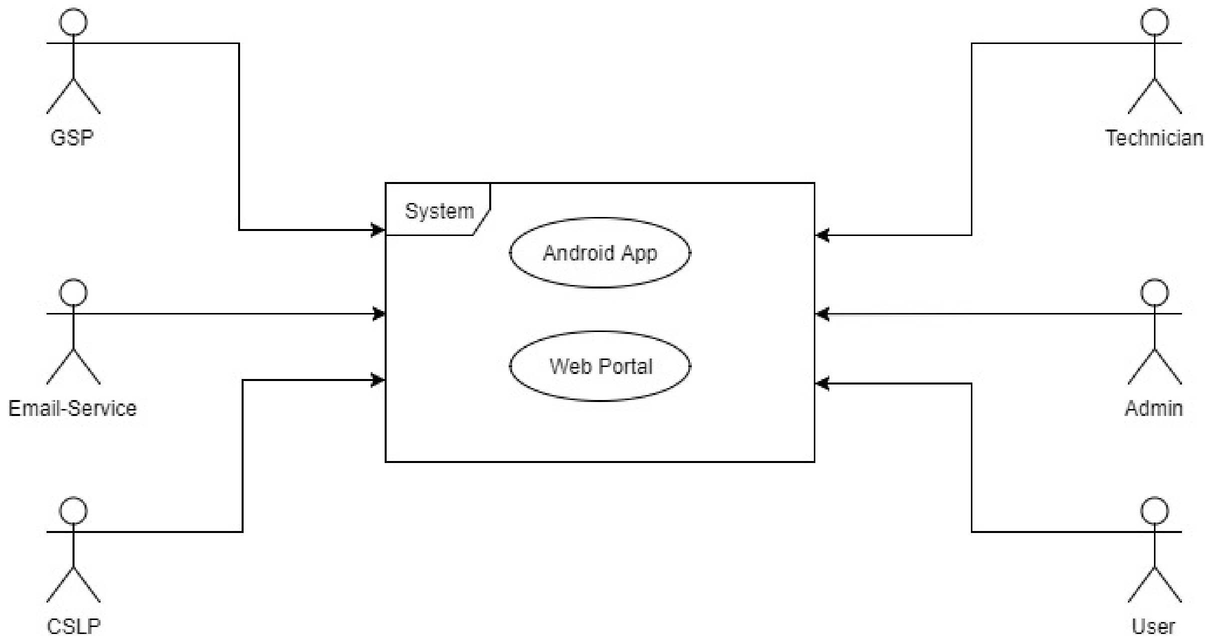



Figure 3.1. Context diagram

the stakeholders?

In general, a stakeholder is an entity with an interest or a role in the system, in our case the main stakeholders are:

1. User: has a read-only access to the server, can view information about counted buildings.

2. Technician: technical personnel: has access to the application and can write to the webserver a new report result

3. Administrator: the administrator of the system, in this case the Politecnico di Torino in the form of the Department of Control and Software Engineer.

4. Developer: the actual developer of the whole system.

The next step is to define what is inside the system and has to be developed and what is outside of it. The tool used to qualify this definition is the context diagram which purpose is exactly to identify actors (entities outside the system, including stakeholders), other interacting systems and the interfaces between the system itself and all the other entities.

Table 3.1. Context diagram table

| Actors | Physical interface | Logical interface |
|---|---|---|
| Technician | Smartphone/Tablet | GUI(Android App) |
| User | PC/Smartphone/Tablet | GUI(Android App), WebPage |
| Admin | PC | Web Service |
| GSP | Internet connection | Web Service (JSON) |
| CSLP | Internet connection | Web Service (manually) |
| Email System | Internet connection | Web Service(IMAP-POP) |

In this case it is useful to roughly show how the interfaces will be implemented. For example, the communication with the GSP will be carried out by a data interface. More in detail: the first step of the communication will be a request (HTTP post) to the GSP with specific parameters, for example latitude and longitude. The response will be a JSON string that will look like this:


```
{
"html_attributions" : [],
"result" : {
"address_components" : [
{
"long_name" : "5",
"short_name" : "5",
"types" : [ "floor" ]
},
{
"long_name" : "48",
"short_name" : "48",
"types" : [ "street_number" ]
},
{
"long_name" : "Pirrama Road",
"short_name" : "Pirrama Rd",
"types" : [ "route" ]
},
{
"long_name" : "Pyrmont",
"short_name" : "Pyrmont",
```

```
"types" : [ "locality", "political" ]
},
{
"long_name" : "Council of the City of Sydney",
"short_name" : "Sydney",
"types" : [ "administrative_area_level_2", "political" ]
},
{
"long_name" : "New South Wales",
"short_name" : "NSW",
"types" : [ "administrative_area_level_1", "political" ]
},
{
"long_name" : "Australia",
"short_name" : "AU",
"types" : [ "country", "political" ]
},
{
"long_name" : "2009",
"short_name" : "2009",
"types" : [ "postal_code" ]
}
],
"adr_address" : "5
"status" : "OK"
}
```


The system will be able to parse this kind of string and collect the data it needs for the user.

Another important thing is to show GUI mock-up if the system expects it. In this project we will need a GUI for the web server and one for the Android application, so in the next figures it is possible to see some easy sketch of the app to be. It is always useful to provide mock-up, because in most cases the client is not a professional in communication design so providing a sketch in which it is explained why the elements a specific position or why they have a particular color is useful to negotiate and decide before starting how the system should look and interact with the final user. But remember that anyway, at last, the client will choose.

Finally, it is clear with who and how the system will interact but still there is no formal definition of the requirements.

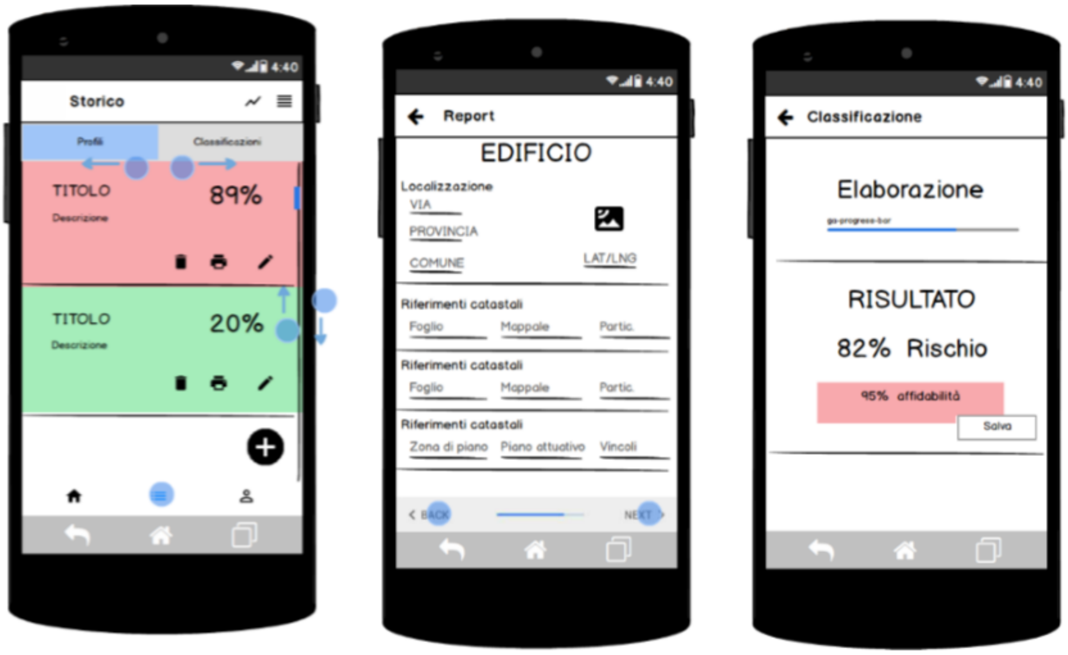



Figure 3.2. Sketches

The next table will list all the requirements of the system, distinguishing F (functional) NF (non-functional) and D Domain requirements. Summing up the most important features that are described in the table the app must be capable of help a technician by collecting very technical data about the building and provide accurate calculations and estimates regarding the subject building.
Without bothering of technical calculation processes (that will be addressed in the appendix) the overall requirements demand to acquire in an accurate way latitude and longitude for the current device position and interact with an internal database to retrieve seismic data. The important thing is that ALL these operations must be available even if the device is not connected to the internet, so no REST call to the database can be made to retrieve data, this means that the database must be embedded into the device (smartphone or tablet). Luckily the database is very small (few mega) and this is not a concern but this will require a manual database update every time the INGV (Istituto Nazionale Geologia e Vulcanologia) releases a new version of it; this manual update would consist in a new release of the application (as a normal update through the application market).
Moreover the app should provide a simple way to insert data, even complex one like the plant of the building, the column shape and must always show the data that are used for the calculation, so the technician can quickly check if the data are correct. The procedure for the seismic evaluation is very complicated so it has been

necessary to simplify the procedure and the calculation by restricting the kind of building and the properties that they can have to carefully and faithfully evaluate them, these concepts are the domain prerequisites at the bottom of the table.
The exports procedures will be 3: email, pdf and upload to server. The upload will send the object report as a JSon string that will be stored on the server, if a report is modified and uploaded again it will replace the existing one. Regarding email and pdf no standard has been requested so these features will not be implemented for now.
Last but not least it must be possible to store media into a specific folder of the app with a specific name, so the user can retrieve them easily and thanks to the several types of possible media, he could integrate with other more specific building data that may not be expected but are still useful to collect.

Table 3.2: Requirements Table

| ID | Type | Description |
|---|---|---|
| 1.1-2 | F | Technician registration |
| 2.1-2 | F | Login/Logout technician |
| 3.1-2 | F | CRUD (Create-Read-Update-Delete) report and report list |
| 4 | F | Automatic collecting of seismic localization data (Istat code - seismic zone) |
| 5 | F | Automatic calculation of closest nodes |
| 6 | F | Calculation of seismogenic data (Ag, F0, Tc*) |
| 7 | F | Seismogenic and spectrum graphs |
| 8.1-2-3 | F | Export report by email, pdf, and upload on server |
| 9.1-2-3-4-5 | F | Storing pictures, audio, video, sketches and notes. |
| 10 | F | View report on web server |
| 11 | F | Seismic hazard calculation |

*(Continues into the next page)*

*(Continues from previous page)*

| | | |
|---|---|---|
| 12.1-2-3 | F | Building section drawing and calculation of perimeter and area |
| 13 | F | Possibility of positioning columns in the designed building plant |
| 14.1-2 | F | Calculation of the column interaction diagram and graph for its section |
| 15.1-2 | F | Calculation of building weight and height |
| 16 | F | Calculation and graph of each limit state in the column domain |
| 17 | F | Seismic hazard calculation |
| 18 | NF | Automatic calculations <1s. |
| 19 | NF | Privacy on sensible building data |
| 20 | NF | Must work offline |
| 21 | NF | Must be quick: extensive use of graphic object to reduce clicks on screen |
| 22 | D | Valid only on Italian territory |
| 23 | D | Italian language only |
| 24 | D | International measure units |
| 25 | D | Concrete buildings only |
| 26 | D | Linear static analysis compatible buildings only |
| 27 | D | At least 3 digits after comma precision |
| 28 | D | All columns are considered identical |
| 29 | D | The column influence area is identical for each column |

Finally there has been some discussion about if the building data should be visible by the public due to privacy concerns. Moreover, there is still no clear intention of

the public administration in Italy to really take care of creating a building census for the whole territory. This remains a utopic object of an experimental project. So together with Professor. Barpi we decided to make the system collecting buildings data from first defined technicians, without ever showing them for now.
For this reason, the requirements no longer valid became: 10 and 19.

### 3.1.1 UML Class Diagram

UML (Unified Model Language) diagram is a way to represent a program in a visual way. It is very useful in software engineering to make communication easier by abstracting from spoken language and giving a handy representation of even complex programming concepts. It is widely used while writing a software architecture because it helps programmers better understand which could be the limit and the bottleneck of the application before starting writing code. It can highlight which parts will require more attention and which other parts could be cut out. If an accurate and complete class diagram is built probably the coding phase would be almost painless and swift.
In the next figure will be shown the first UML class diagram written for the system, the most important points are:

1. The domain area includes all the domain classes, the core of the system. These classes holds the knowledge of the report and its data. The domain classes have a 1:1 connection to the report record which is the ID that makes the specific report accessible.

2. The domain is strictly depending on 'Normativa' meaning that if the laws about seismic vulnerability change the domain must change accordingly.

3. Each report may contain a list of report media.

4. Each technician can access the previous reports using a history section.

5. For each report record exist a report result an export operation that could be of 3 types: PDF print, Email, Web Portal upload.

6. The web server will hold a user table and a report registry.

7. The outside actors interact and modify the domain data.

This was the first attempt of UML Class diagram, with just a general knowledge of the requirements, it lacks the business part interacting with the part of domain that can be modified by the user and totally missing the view model part in which would be defined how the data should be represented but it is a good snapshot on the functionalities and the general form of the applications. It will be used later in the architecture part as a starting point to define, in a more detailed way, which are the domain objects and properties, how the data are represented and how the user will manipulate and view the data.

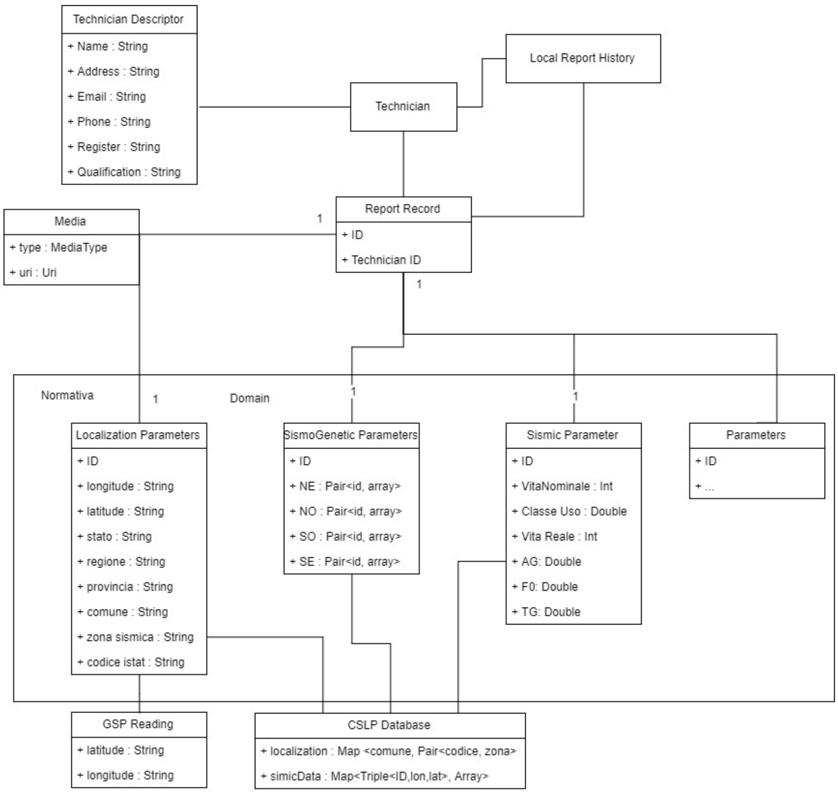



Figure 3.3. UML Class diagram, part.1

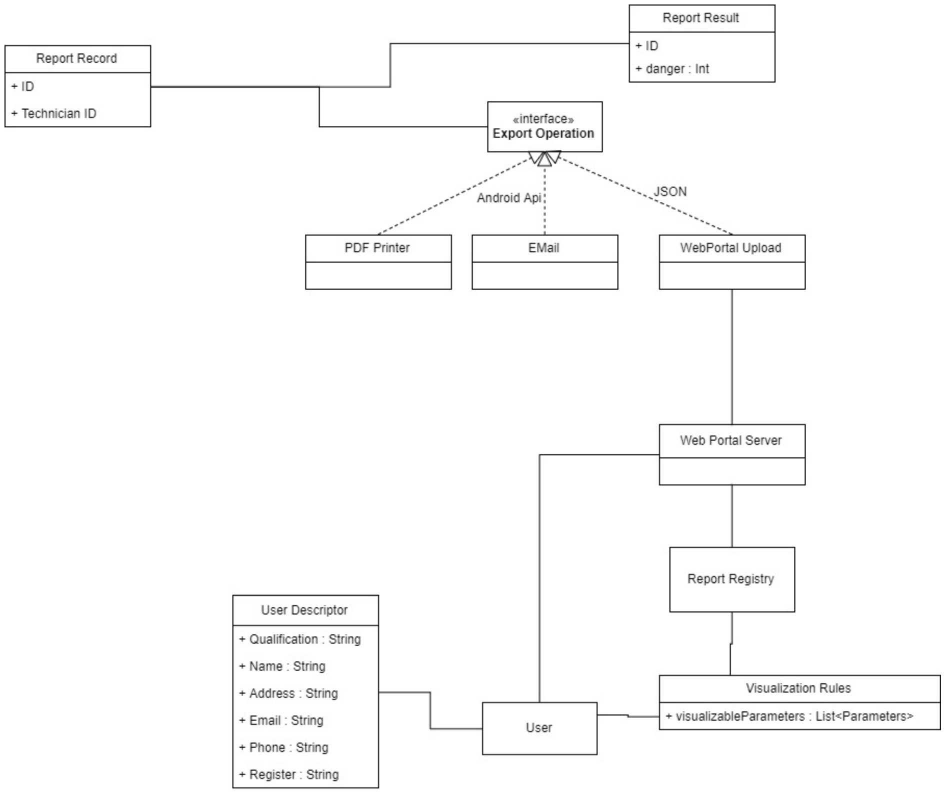



Figure 3.4. UML Class diagram, part.2

### 3.1.2 Use cases and scenario

A use case is a written description of a specific user task divided in single steps. Is a list of action AND events that define the interaction between the actors and the subject system. The most important use cases are usually the stakeholder goals, and the analysis of them is made easier with the use of predefined templates, using design scopes and goal levels to summarize all the outputs and features of the system together with preconditions and extensions that help to answer the usual question: How should the system behave in this situation?

Table 3.3. Main success use case

| | |
|---|---|
| Nome | Filling report on field |
| Goal | Main goal |
| Level | Technical objective |
| Objective and Actor | a Technician wants to fill a report about a building and save the data onto the portal |
| Prerequisite | The technician is already registered on the portal |
| Scenario | 1. Techincian logs in.<br>2. Technician starts a new report.<br>3. He gets the current position using GSP from the app feature.<br>4. The application uses the GSP data to inject the data read from CSLP database.<br>5. Technician fills all the requested data.<br>6. Technician adds medias.<br>7. Technician saves the report and choose to upload it. |
| Extensions | 1. 2.b Technician modifies an already existing report |

In table 3.3 the main success use case for this system is explained step by step. This is the main goal of the system: make possible to a technician to quickly fill data about a building and send it to the server as an element of the census. Some applications of it can be found in the scenarios tables. These elements provide a real-life context example of how the system will behave to better understand why the app behaves in a certain way rather than another one. For example, in the first scenario, 'ON-FIELD REPORT, AFTER AN EARTHQUAKE' another possible choice could have been to lock the application if internet wasn't available, because the GSP wouldn't provide the location, and consequently it would have been impossible to read from the server the data about seismic action in the area, making the report, in fact, useless.
In the successive two tables (3.4 and 3.5) instead are shown two possible scenario, contextualizing the main success use case showing which should be the feature the app should be available and which would be the limit of the usage of it. In the first scenario the condition is the impossibility of using the internet connection to retrieve data, in this case the application should provide in any case a way to input data to continue the procedure of evaluating the building vulnerability. The second scenario is about a feature that the application should provide to correctly reach the main success scenario: the possibility to edit an already filled report, changing wrong data and re-evaluating the building without creating a new one from scratch.

**SCENARIO 1: ON-FIELD REPORT, AFTER AN EARTHQUAKE**
**Preconditions:** Technician already registered and logged. Internet connection NOT available during the analysis, available only few hours later.
**Result:** The report is correctly saved on the device and later uploaded on the server.

Table 3.4. Scenario 1

| Step | Description | ID Requirement |
|---|---|---|
| 1 | Technician opens a new report | 3 |
| 2 | Technician inserts data regarding the position manually | 14 |
| 3 | Technician adds pictures | 9 |
| 4 | Technician inserts all parameters and reads the result | 3-11 |
| 5 | Technician few hours after, when connection is available, uploads the report on the portal | 8 |
| 6 | Technician export a pdf about the report | 8 |

**SCENARIO 2: MODIFICATION OF AN EXISTING REPORT**

**Preconditions:** Technician already registered and logged. Internet connection available. Some new information about the building come available.
**Result:** The report of the specific building is modified accordingly to new data.

Table 3.5. Scenario 2

| Step | Description | ID Requirement |
|---|---|---|
| 1 | Technician views all available reports in the app and open the one relative to the subject building | 3 |
| 2 | Technician modify the report with the new data | 3 |
| 3 | Technician adds audio where explains which are the modifications | 3-9 |
| 4 | Technician views the new results returned by the app | 11 |
| 5 | Technician uploads the report on the portal | 8 |

## 3.2 Implementation

This chapter will describe the steps of the Android and web server implementation together with the explanation of the main issues occurred during this phase. The writing of requirements and architecture took one week while the development and testing phase took roughly two months. Each sections of this chapter will show in detail how the app and the web server were engineered, a brief explanation of the software architecture chosen to be implemented and the algorithms used behind scenes and even the explanation of why each single UI object appears in a position rather than another one.
Regarding the testing phase, it has divided in two different phases: the first contemplated unit test on the single business classes that interacts with the domain data. This is very important, especially if the software development process is TDD (Test-Driven Development), because it allows to write test cases before coding the real thing using stub classes. At the end of the process the real implementation will have to pass all the test that were written during the development.
The second phase was contextual to the first and was carried out by Denis. His duty was to manually try the app reporting possible bugs or suggesting changes in the UI for a better user experience. He would be a potential user of the app and his feedback has been a great assess during development.

The steps through the implementation chapter follow the flow of the real development, that followed the process of DDD (Domain-Driven Design). First were written the domain classes, meaning the classes that contains the data that will be inputted by the user for the building calculations. Second were written the ‘interactor‘ classes. Their job was to access the domain classes and change the state according to outside inputs. This is the main core of the application, it is totally abstracted from the view model part and can be tested. Last to be developed has been the UI part, it sees only the interactors that expose public methods to change the variables of the report, in this way the UI layer doesn,t really know what happens when a state changes, it only knows if the changes succeeded and what is the result to be displayed.

### 3.2.1 Kotlin

The Android application has been developed with the brand-new language Kotlin. In the last year [2] this language has officially become supported by Google for Android development, it means that every app that will be written and has been written in the past in Java, can be ported to Kotlin. Why is this language so special? The Kotlin project actually was born in 2011 by JetBrains, is a statically-typed programming language design to interoperate with Java and it runs on JVM (Java Virtual Machine) as well. Syntax is not the only thing different from Java, being an opensource language born from an industry and not an academic language, it solves many problems faced by programmers in real life development. Some Java issues and limits are overcome and it actually gets very close to C # [3]. The most important new features exposed by Kotlin are:

1. **Extension functions:** It is possible to extend a class without having to inherit from it nor using a decorator. Extension properties are also supported.

2. **Lambdas:** Lambdas are anonymous functions, meaning function without a name (identifier). This allows to write much less code and to inline functions where is needed, moreover if the function has only an argument as parameter, it can be referred with the type word ‘it‘, without using the usual ‘name -> syntax‘.

[2] Announced on May 2017, during Google I/O

[3] Only parts of it are opensource since few years

3. **Null reference controlled by the type system and null safety:** As referred to ‘The billion dollar mistake‘ [4] Kotlin tries to eliminate the danger of null reference using !! and ? operators and optional Type? types to help the programmer to know what can be null and what cannot.

4. **Smart cast and Pattern matching:** using ‘is‘ checks the compiler is smart enough to explicit safe cast variables to checked type; of course it may not work if the variable is changed from declaration to check (by a lambda for example).

5. **Type inference:** In most of cases is not required to explicit the type of a variable inside a method (using ‘var‘ or ‘val‘ if the variable is mutable or not).

And actually much more. The fact that Google supports it for Android development means that this language has a lot of potential and with all these new features it allows to code an Android application in a much faster and safer way than usual, if one can take advantage of all the potential offered. Moreover, just very recently JetBrains announced native support for Kotlin by using the CLion IDE (still beta version) that moves Kotlin apps out of the JVM, allowing to deploy a Kotlin app not only on Android devices [5]. Other functionalities and features available will be explained in the next sections in the code provided samples.

### 3.2.2 Clean Architecture overview

The main object of software architecture is the separation of concern as the good old ‘divide et impera‘ paradigm never gets old. The main pros of this architecture are the ease to write, handle, maintain and even debug a program that is subdivided in layers because the programmer can understand at first sight if the reported bug is actually just a user interface bug (a string formatting non-correctly written for example) or a miscalculation in the data (that would suggest a problem in the function that calculates a result). Nevertheless, in this way, the development of the different layers 3.5 could be handled by different teams, once the contracts as known as interfaces, are chosen and, most important, a change in a layer doesn't usually affect the others layer. The main cons on the other hand are two: first it takes

[4] Hoare, Tony (25 August 2009). "Null References: The Billion Dollar Mistake"

[5] https://www.techworld.com.au/article/629643/beta-jetbrains-ide-moves-kotlin-apps-jvm/?fp=16fpid=1

quite some time to carefully define business entities and use cases and second the dependency inversion principle (explained later) may not be really intuitive at the beginning.
The concept at the base of this architecture is 'The Dependency Rule' stating that source code dependencies can only point **inwards**, this means that nothing in the inner circles must know anything of outers circle and it is represented in 3.5 as the inwards arrows. In this way the outer layers become dependent on the inner layers and use the classes and method that they provide, but the opposite is not true. The layers are four: Entities, Use Cases, Controllers and Frameworks in the next paragraph will be described these concepts as abstracts while in the next section will be shown how these concepts has been implemented.
The entities layer contains all the domain object of the application, it can be data structures as well as data classes with methods and variables, it depends on the purpose of the application. In general, all the data contained in this layer are not subject to change when the external world changes (think about a change in a navigation page, it doesn't affect the domain data), and this is the first most important layer and is the first step into a DDD process.
The second layer is the Use Cases, it describes and implements each specific busi-

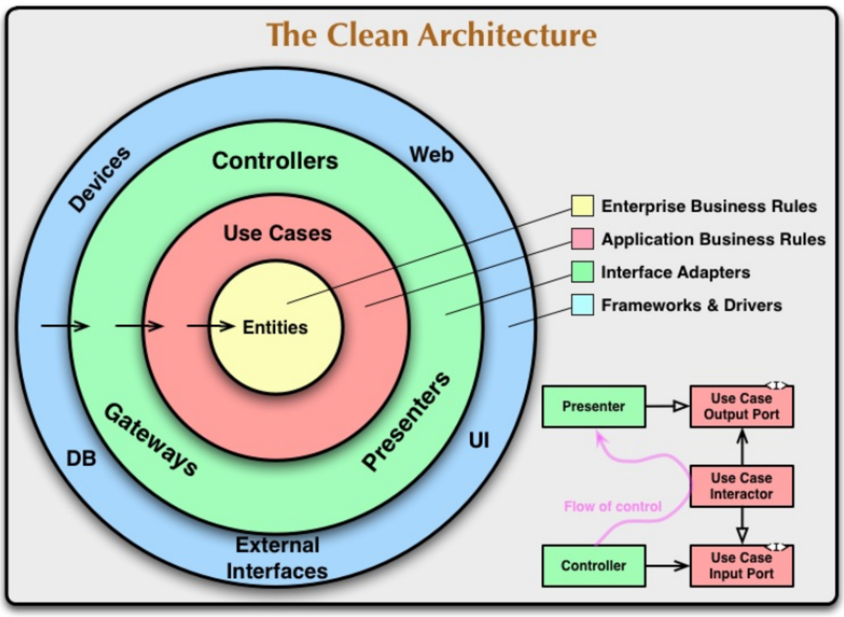



Figure 3.5. Clean architecture layers

ness rule of the application by manipulating the domain entities. If a behaviour or a calculation of the applicant must change, this is the layer that will be affected and, most important of all, if something external changes like the database or the

UI graphic the use case won't be affected, it will just keep working as usual. With use case in our application we intend usually a method that contains all the logic to complete a user action request.
The third layer is called Controller because it responsibility is to convert the data from the business format of the application to the format requested by the most outer circle. A very simple example is database accessing: if we consider an SQL Database, the database itself is in the outer circle (since is an external component), the adapter knows SQL and is responsible to query and write onto the database when a use case requires it but, the use case itself doesn't know anything about the SQL nor the database, it will just implement a method called 'save' and will call a method of a database adapter class that will do the save for it.
Last circle contains all external component that really don,t need much code, the only code that will be written is the one that will attach it to the inner circle to use the adapters contracts.
In the bottom right corner of 3.5 is shown a simple operation flow from the presenter to the controller, let's make it a less abstract with an easy example: the presenter knows that something has change in the UI and has to execute an operation on the database or into the web. The presenter could directly call the controller, but the business entities won't be correctly updated, so it has to use the use cases to have the right access and complete the operation correctly. But the use case cannot directly call the controller to execute the operation (for the dependency rule). This apparent contradiction is resolved using the dependency inversion principle that uses interfaces to abstract implementation hiding the real call to the controller. Another way would be to use events to keep synchronized the outer layer with the inner ones, but this is expensive in terms of time and code and could lead to a lot of bugs if not used properly. The only thing that can pass through interfaces are usually simple types or DTO (Data To Object) class made for the purpose.

### 3.2.3 Domain data structures

In this section will be detailed all the business entities and will be explained why they were implemented in such data structures with the hope to clearly show more specifically which operations has been requested for the seismic hazard assessment of the building. The domain data structures that compose the business entities layer are classes used to represent significant information about the building and its location.
The central domain object of the application is indeed the building and its properties, however the purpose of the application is to fill a report about the building so the attention must switch to this object. A report has a unique ID that identifies it onto the local database, a User Identifier that identifies the user who made it and

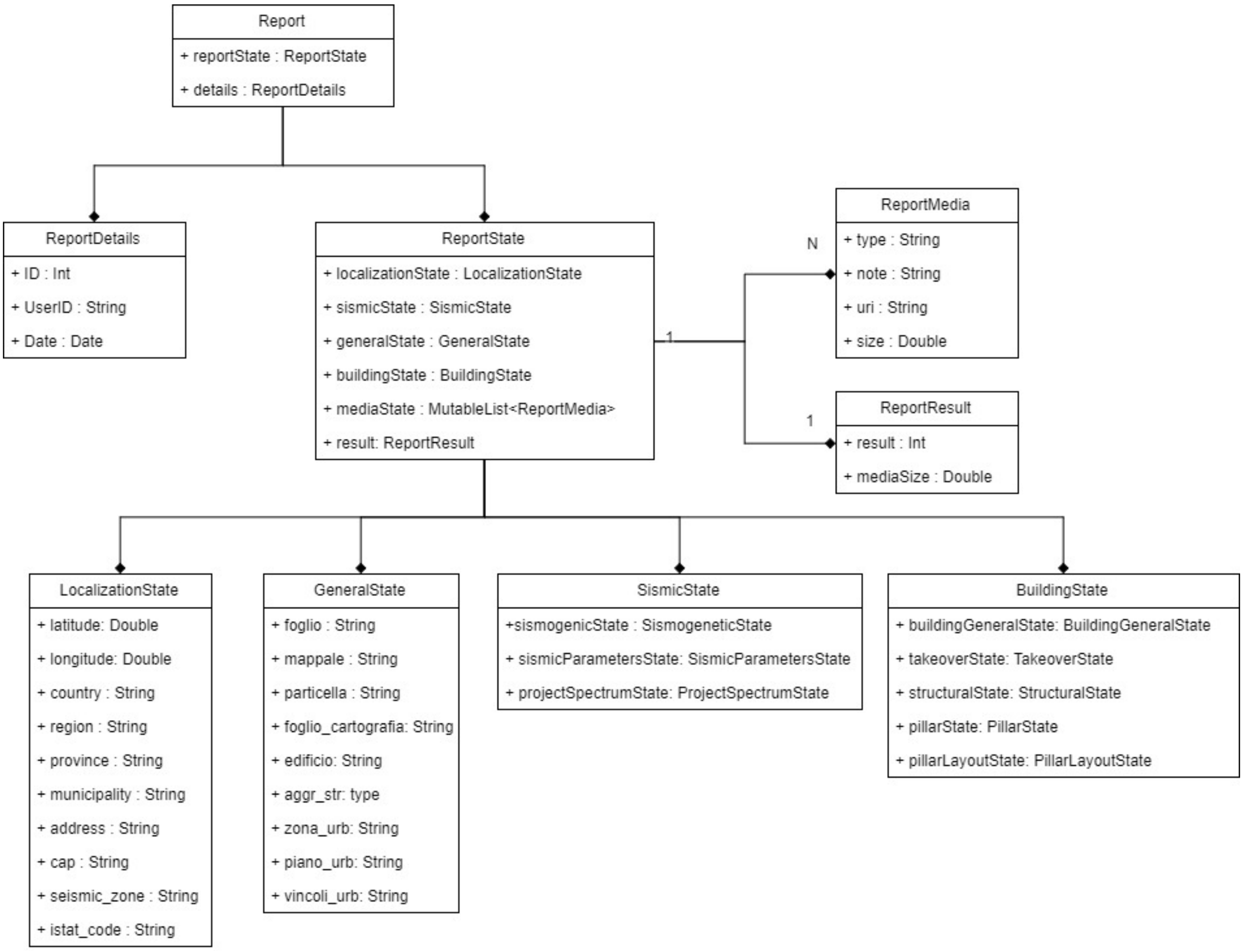



Figure 3.6. Report business class

a Date of creation. These are the only data that the user cannot change. These are the invariant data and are represented into a read-only class called ReportDetails. Another class called ReportState is responsible to collect the data that can be changed at runtime by the use cases layer, in particular the main actor of these changes is the user, even if in some cases the GSP and the CSLP are responsible to automatically change data. The information that can be inserted can belong to mainly four categories: location, seismic data, general data, building data; each of them is represented by a data class.

The figure 3.6 shows how the report is organized; while localization state and general state (written in Italian because are data strictly bounded to Italian laws) are pretty easy class to understand seismic state and building state contain a lot

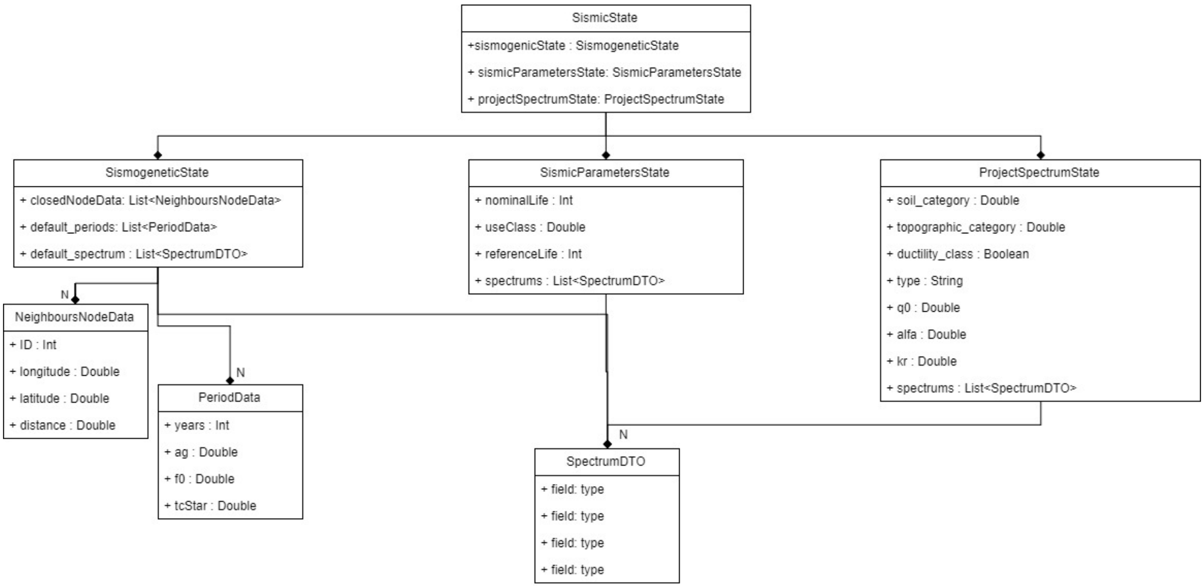



Figure 3.7. Seismic state business class

more. These last two classes must contain every available detail about the seismic characteristics of the location and the shape and constructive details of the building, like material, number and composition of columns.

In particular in figure 3.7 and 3.8 are visible the UMLs describing the components of each of these two composite classes and the properties they contains. The seismic state class owns data regarding the sismogenic, the seismic parameters and the spectra. The first kind are information provided by the CSLP in .csv files that using year and acceleration parameters (ag, f0 and tcStar) describe how intense can be an earthquake in a particular area. By querying the database, looking for the closest nodes and their parameter, it is possible to estimate the acceleration data of an area having only the coordinates of it. The result of this interpolation is a function that describes the acceleration of the terrain in the first four seconds of an earthquake. By crossing this function with the seismic parameters of the building, it is possible to determine four different functions, one for each so-called limit state (a condition that if exceeded can jeopardize the building) that depend on the reference life of the subject structure. These limit state describe a condition that the building has to satisfy: SLV and SLC are associated with the carrying capacity and if exceeded may cause a human life loss (or the collapse of the building. SLO and SLD are associated with the deformation of the building that can or can't be reversible. Finally, spectrum data add a level of accuracy to the previous limit

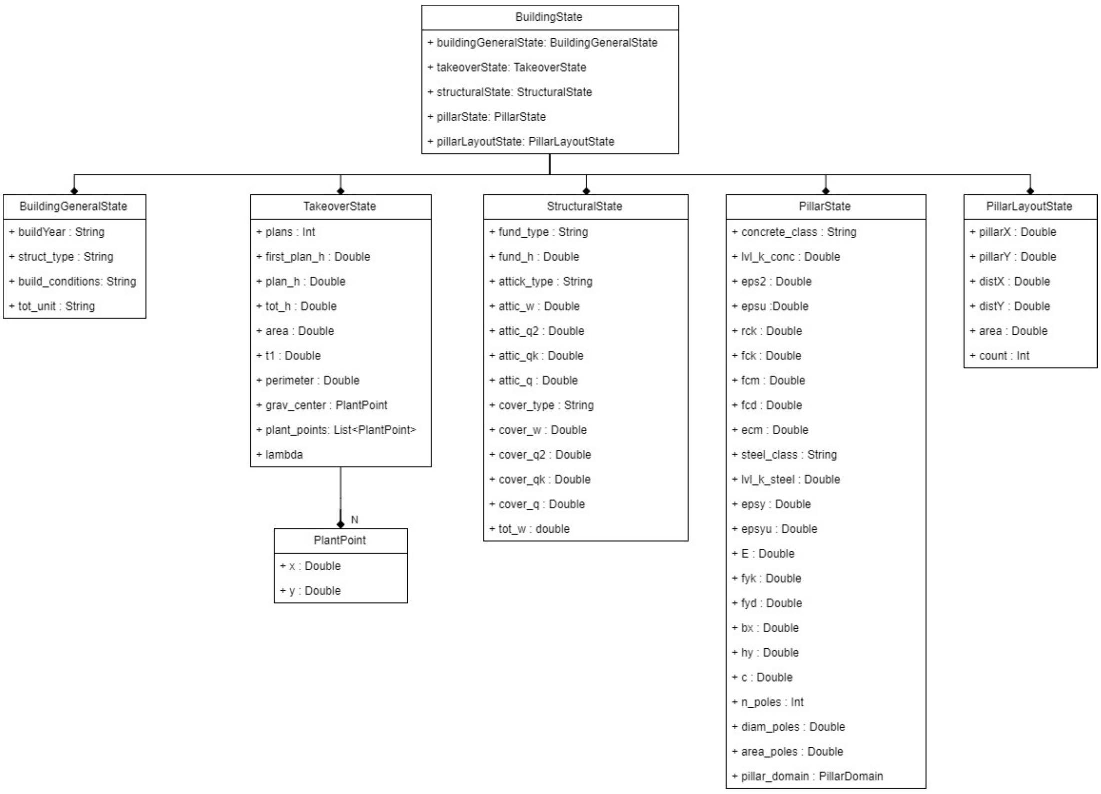



Figure 3.8. Building state business class

state by modifying them according to the formula in B, these parameters contain information about the soil on which the building has been built and some general information about shape, regularity and ductility of it.

The building state on the other hand contains all the specific information of the building, some are intuitive like year of construction or type of building, but the majority are very technical information. The first class that forms the building state (from the left to the right of 3.8) is building general state and contains only generic information. The second one is takeover state: it contains all data that are collected during the measurement activities, both internal and external. As a matter of facts, it contains the height of the building and for each plan and total and the shape of the plan of the building, inserted point by point in a two dimensional plane. The structural state contains the data regarding the weights of the single components of the building together with the total weight of it. The column state holds all the parameters of the single columns, elasticity coefficient, material type, and more technical ones. All these data allow the calculation of the column domain, a graph that shows an area in which the column can resist stress. Last class is the column layout that describes how the pillar are settle in the foundation and the total number of them.

### 3.2.4 Assumptions

Domain structures have been thoroughly explored and the purpose of the application is clear. However the provided functionalities doesn't cover all the possible combination of building and location that may happen during a seismic evaluation. These limits are listed in the requirements table 3.2 and can be identified by the type D (Domain), requirements 24 and 25 are technical limits of the seismic vulnerability used in this project. The 24 is clear and states that the evaluation is accurate only for building made of concrete, while the 25 is a bit more complicated. It states that the calculations are valid only if linear static analysis conditions are respected: value T1 must be less or equal to Tc, the total height of the building must be lower or equal to 40 meters, and the building itself must be regular in plant and height. T1 and is a value that correlates the vibration of the building to the height and Tc is the time in which the acceleration during an earthquake drops. Regularity in height and plant means that each plan must have the same shame considering its horizontal section. All these conditions must be checked at runtime and an error should be displayed to the user if at least one of the condition is not respected. Such logic is necessary to not mislead the user that could use invalid data during the analysis. All these conditions may restrict the usability field of the program itself but the complete evaluation of the totality of possible buildings on the Italian territory wouldn't be any more an academical project thus going out of this project topic.
Another assumption is that the server will only offer three easy basic functionalities: registering and logging in the user using only encrypted password for privacy and receiving a JSon string identifying a report sent by a technician that will be stored on the database of the server itself. The server will be deployed into an Apache server of the computer engineering department at Politecnico di Torino in a temporary way, until some decisions about the future of the census project will be made.

### 3.2.5 Interactors

The interactors are the classes that own the domain data structure and modify them on mainly user requests. The Android activity used to input the building and site data is mainly a stepper activity. Each step contains a part of the domain data of the whole report, divided by category and ordered trying to follow the normal flow of an onsite evaluation. The first step for example contain the localization data, that can be input manually or using the features provided by the GSP that through the internet connection and GPS signal can retrieve the current location of the device. In this case the interactor is in charge to get the user request (for example to acquire the location automatically), modify the location state instance of the report, that is the domain class that owns the location data and the update the user interface to display the current and newly updated domain data. The flow of these operations can be seen in figure 3.9.

The validation of the data is responsibility of the user interface that must know

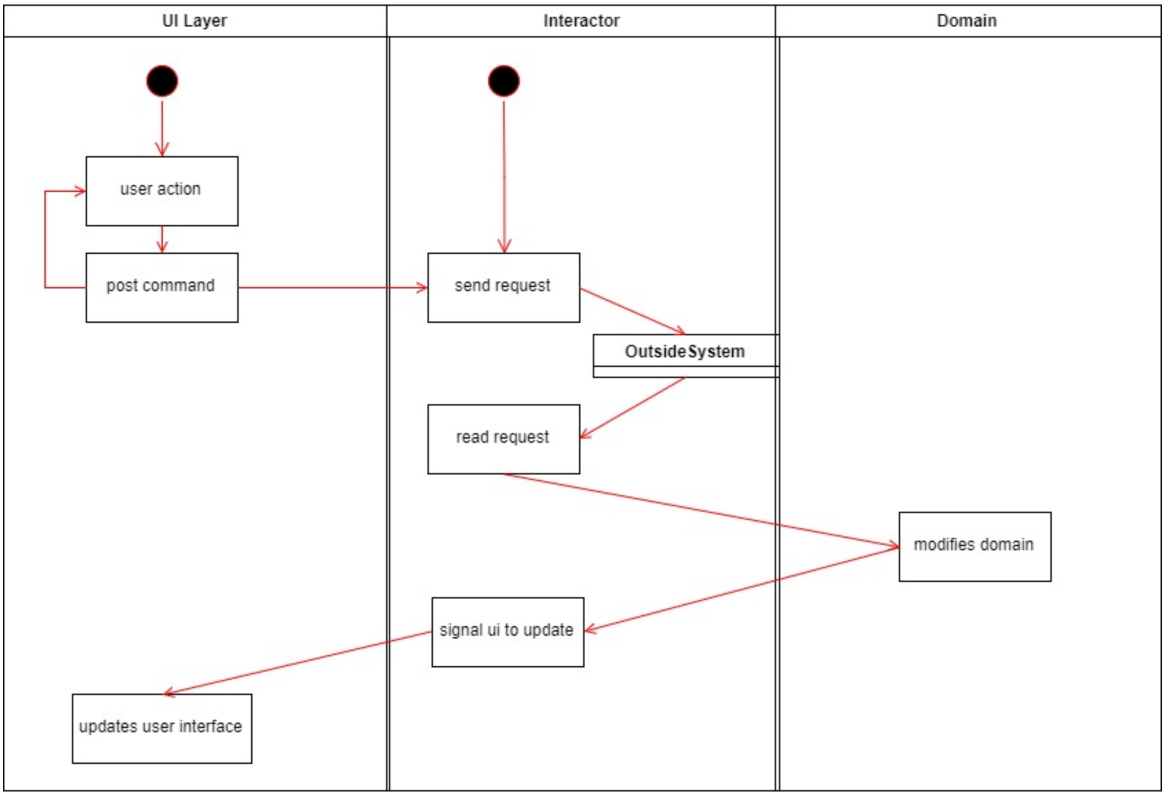



Figure 3.9. Flow of the localization interactor

which data are mandatory, or the right range of usability. Once a page has been confirmed, if all the input is valid the user can step to the next page and the data of the previous phase will be finalized into the domain class by an interactor that will read the data from the user interface and modify the domain class accordingly. An important thing to point out is that in Android the Fragment class is a Presenter, that means is a class that knows how to draw itself and represent on the UI layer. If no complex action is necessary, nor validation is useful on the represented data the interactor can actually be omitted because it would just add useless complexity

to the classes code. Is the case of the second view of the report that only contains optional data indicating some building data that can be read by the Italian registry. The third fragment is a bit more complicated since it has to compute on open the closest nodes position from current location to correctly represent the seismic response of the area for default limit states. This operation is implemented by an interactor and a helper which function is to read from the local database of Italian seismic data the closest measurements and interpolate them with the current location based on distance A. Once the calculation succeeded the result will be displayed to the user using a list indicating the enclosing four nodes with their distance, id and coordinates. Together with the list, that is needed to check if the closest nodes are correct, a graph will be showed to see the trend of the default seismic response for the area and, at the bottom of it, the interpolated data for the area.
The fourth and fifth fragments use and interactor as well to get the default seismic state from the domain previously calculated and modifying it using the data that the user can input in the two views. At the end of the fifth fragment will be visible a graph showing the result of seismic response and all its calculated parameter.
While sixth fragment is again an optional container of data, the seventh allow the user to input the height of each plan of the building and its plant and modify for the first time the building state of the report. An interactor is responsible to keep a temporary list of vertices that can be manually modified by the user to draw the plan using the opposites buttons to add, remove or close the graph. Once the user confirmed all the vertex, a graph will be again shown to display the lines which make up the plant, giving the opportunity to check if the input points are correct.
The next fragment contains the data about the weights of attic and coverage, that will be used by the interactor to compute and store on confirmation the total weight of the building, using the previously input of building height. The last two fragments are responsible of the columns data: the first contains the data physical data of each single column like construction material, height and internal shape of iron bars, allowing the calculation of the pillar domain graph, that represents how much weight the column can hold. The latter contains the layout of the column on the building plant and allow the user to modify whether a column is present in the real location or not. All the interaction between the user interface and the data pass through the interactor which is the only one in charge to modify domain data.

The last fragments contain a summary of all input data not allowing modifications anymore. The very last fragment before confirmation shows the result of the evaluation, crossing seismic location data with the physical properties of the building and its columns.
Another feature of the report activity is the input of media data. For this operation another interactor is responsible to catch the user request from the specific button and ask another activity to be built to input a file that could be of 3 different types: mp3, mp4 and jpg. Once the picture, video has been taken, not has been writer

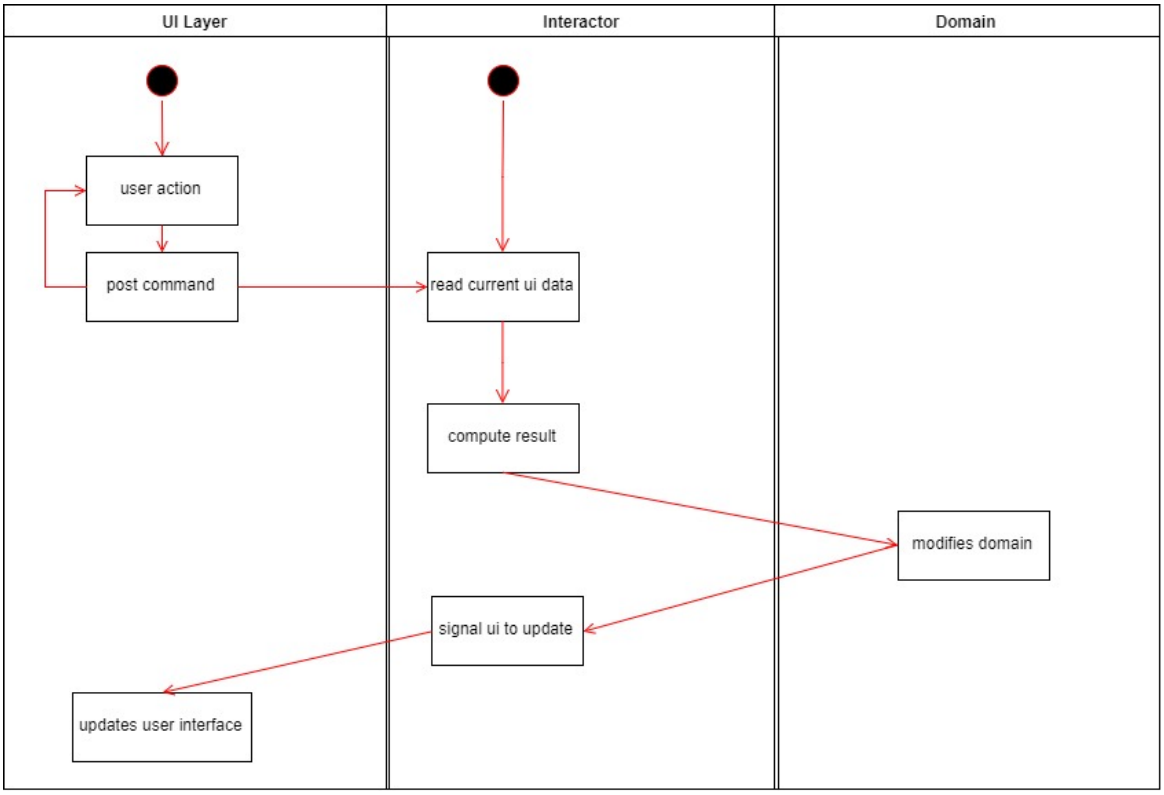



Figure 3.10. Flow of the column domain interactor

or sketch has been drawn, the interactor will store into the report domain a media instance that contains the type of the media and its uri that make possible to identify the file onto the Android filesystem. The flow is very similar to 3.10 with the difference that the outside system is in this case another activity.

### 3.2.6 Material design overview

Material design is a design language first introduced by Google [6] to create a consistent user experience across all devices and screen sizes. It is a design language that has been adopted not only on Android devices but on web applications as well allowing a user not to be stressed out about new types of interaction with the software, in fact the same gesture on different application that use material design will cause the same reaction by the program. The main principles of Material Design

[6]Google announced Material Design on June 25, 2014, at the 2014 Google I/O conference

are:

1. **Material is the metaphor:** because graphic objects maintain their properties, especially light, surfaces and movements reflect the physic of the real world without been strictly bounded to it.
2. **Bold, graphic, intentional:** typography, grids, space, scale, color create hierarchy and focus.
3. **Motion provides meaning:** motion and feedback are appropriate and clear.

Thanks to Material Design the user experience can be clearly defined giving the developer a great advantage while coding because he has just to follow general guidelines that describe how the motion should be made, how the information should appear, which component should be used to more effectively describe and give meaning to concepts and objects.
Following Android guidelines, the best component chosen to describe the procedure of report filling has been the stepper. It is composed by a series of swipable pages that contains different but correlated data. Each of these pages has a title that indicates the generic information that will contain, together with a number that identifies it. At the bottom are present two buttons to navigate forward and backwards: when a page is correctly completed the user will be able to step to the next page, if some input data are not valid or missing, a message together with a red alert will be shown to indicate the user the parameter to fix. The backwards navigation is always possible. Some specific page related action is provided at the top using action button. A floating button at the bottom right will provide access to media inserting: this is the standard procedure but since the media inserted could be of different types, the floating button morphs into a bottom bar that allow the user to choose which kind of media must be inserted (video, audio, sketch note, picture); this bar will disappear if a scroll or a loss of focus happens to effectively show data that may be hidden under.
Colors have been widely used as well, together with graphs to give a better feedback to the user regarding the inserted data. Colors provide meaning to the result as well because the more the building is in danger the more the result will be red, the usual color associated with danger. Graphs instead have the double function to group complex data into a single element because the user can understand the course of the function and check its correctness in a simple glimpse of an eye. The plotting ability of graph has been used to also represent the plant of the building and to show where the column is positioned in it, together with the touch interaction it is also possible to switch on and off a column in the case it should not be considered nor is present in the building.

### 3.2.7 Presenters

This section contains a detailed description of each fragment of the report, together with some sample screenshot and the explanation of the design choices that was taken during the development phase to provide a better user experience and to quickly complete the process of building assessment. At any time the user will be able to step back and change a parameter that may be wrong. The floating bottom at the bottom right corner (a close position to the user's thumb) if pressed becomes a bottom bar that disappears on scroll and that allow the user the input of media files like notes, sketch, audio or video. These files will be listed at the end in a summary fragment.
The first step of the procedure requests the user to input the current location. This

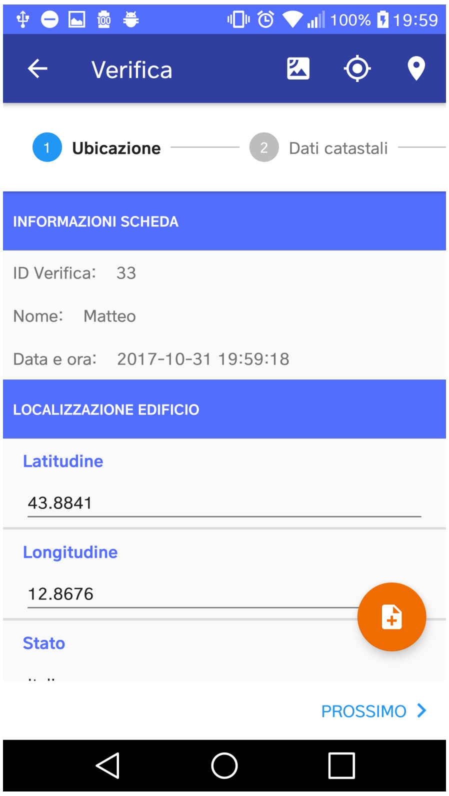



Figure 3.11. Localization fragment

operation can be made in several ways, according to the requirement stating that the application must work normally when the internet connection is absent. The most difficult way to insert coordinate is manually, in this case the user must read the current coordinate using another tool and manually digit in the provided field the latitude and longitude.

Another way that doesn't require internet connection is to use the central button at the top of the bar, that will automatically input the last coordinate read by the device gps system, this action though will input just the coordinates and not the other data like country, region and municipality. The easier method to fill all the fields in this fragment is to input an address or picking a place on the map. These actions can be made through the first and last button on the top of the bar

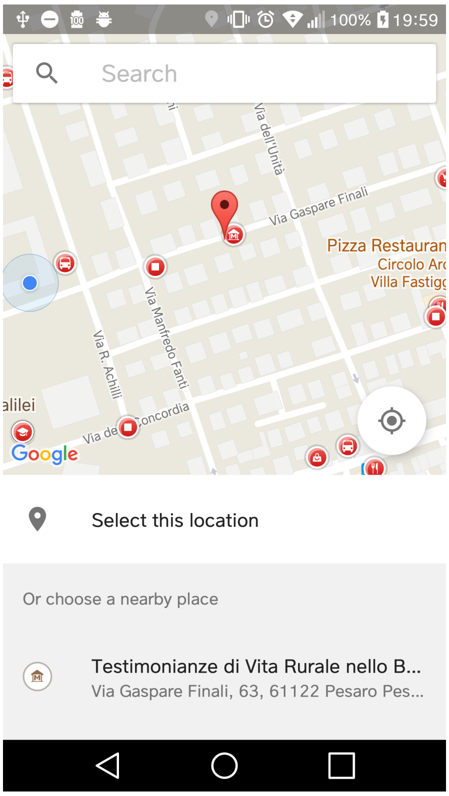



Figure 3.12. Activity to acquire position

but need internet connection to send a POST request to the GSP and receive all the information of the current location. The only information that the user must input manually is the CAP. The parameters 'zona seismica' and 'codice istat' are data read from an internal database, provided also by the CSNL that indicates, for each municipality, these two parameters. In fact if the user choose to input manually region, province and municipality, the program will automatically show the suggestions for the current inputted digits to make things quick and error free. The suggestions are also read from the database, in this way it is very difficult to input a city not present into the database.


Figure 3.13. Land register fragment

Once all the localization parameter is inserted the user can step to the next fragment that shows data regarding the Italian registry (catasto). This information are not mandatory for the seismic building assessment so the user can leave the information empty if at the moment are not available and step to the next fragment. When a web system for the Italian land register will be available this procedure can become automatic.

The sismogenic fragment will show a list with the closest nodes for the current location, their correspondent distance and ID in the database. This also is just an information fragment, in this way the user can check if the nodes calculation succeeded and can see the default spectra for the default return time in the area after the interpolations. Each return time function has a different color, without any significant meaning, just to help the user identifying them.

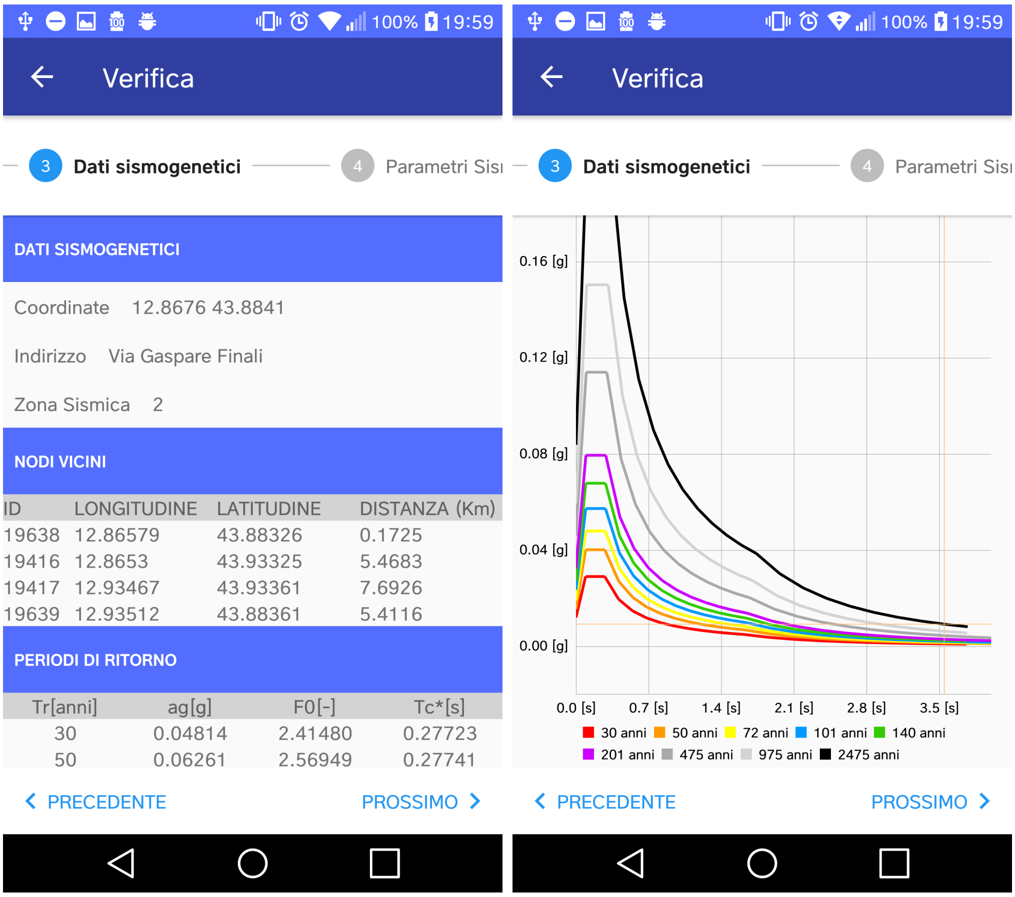



Figure 3.14. Seismic hazard fragment

This next fragment allows the user to input the nominal life and use class of the building that together define the reference life. It will be used to calculate return times and to show the interpolated acceleration function for the current position. Since the nominal life can only be 30, 50 or 100 the normal edit text has been replaced with three button that share a state, meaning that only one can be active. In this way the user can just click on the nominal life without any further inputs. This choice has been taken further in the app every time the user would have had to choose between two or three options, a greater number of buttons in fact would have made the graphic representation of each button unclickable and too small for material design. At the bottom of the page the user can find a list of all calculated parameters for the current graph. Each limit state has a different color that indicates the importance of them and the reference calculated life can be checked through a label right above the graph.

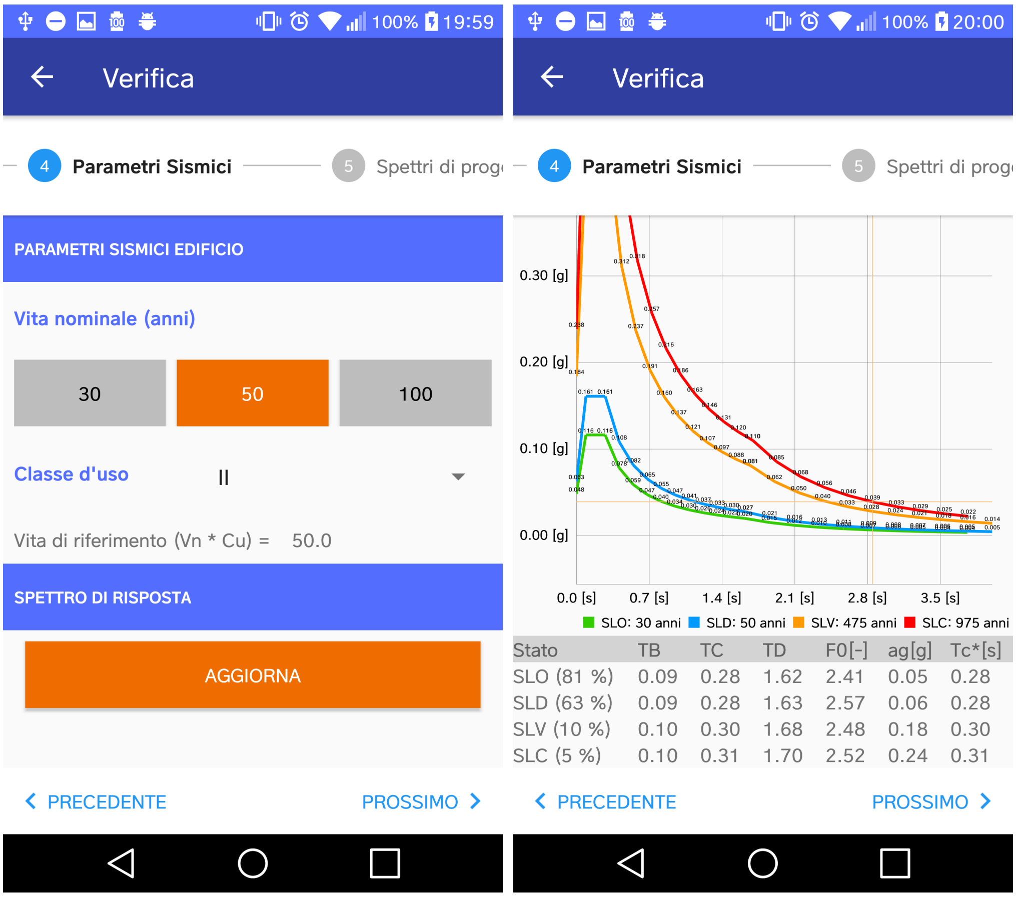



Figure 3.15. Seismic parameters fragment

This is a mandatory fragment since it defines the real spectrum function of acceleration of the site. It allows the user to input very specific data regarding the building and the site itself using spinners to make things faster and error free. The top section contains data regarding soil category and topography that can be changed through a spinner in a fast and mandatory way while the second section contains data regarding the general structure of the building. These data are tightly bounded to each other and a particular value of 'topology', for example, make $\alpha$ disappear, since it won't be used for the calculation in particular cases.


Figure 3.16. Spectra fragment

These data allow the calculation of the parameter ‘Q‘ (structure factor for [15]) whose value can be checked into a label. The checkbox ‘Regolare in altezza‘ is just a placeholder since it is a functionality not yet implemented. The refresh button will show the updated graph after the calculation explained in B.
At the bottom of the fragment, after the calculation succeeded, are shown all the data used in the calculation of each limit state, a bit more of what Spettri does, but a bit less since is not showed the list of point that make up the graphs as the excel does.

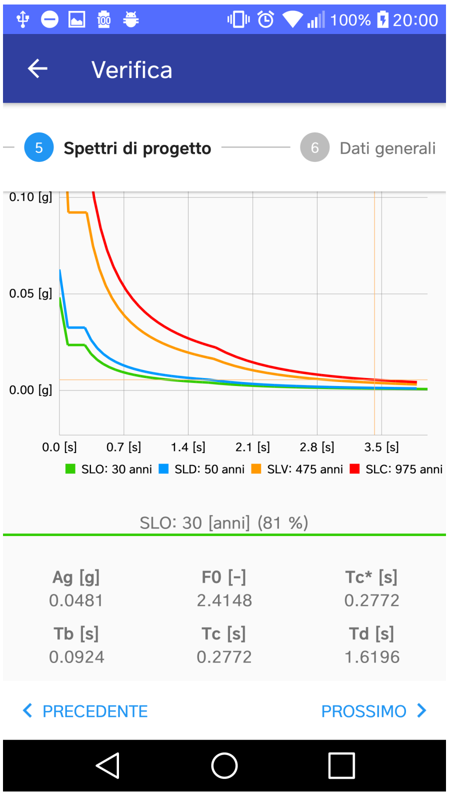



Figure 3.17. Take-over fragment pt.2

The general data fragment is again an optional fragment with just spinner to input generic data regarding the building. These data are not used for the process of seismic hazard evaluation but can be useful to the technician involved in the process while compiling a report on the building.


Figure 3.18. General data fragment

The takeover fragment is one of the most complicated one. It allows the user to input the plant of the building and its height. Regarding the height it has been sufficient the usage of a spinner for the number of floors and two edit text for the height of the first plan, that is usually different, and the generic height of a plan that, since all plans must have the same height under the regularity requisite. The input of the building plant is a bit more complicated. It requires the user to input by hand, one by one, each consecutive vertex of the plant, starting by a reference point that will be represented on coordinates ‘0,0‘. It is very similar to what in reality happens during the measurement of building, making this process very similar to the real-life process, and in a future, could be automated if the measurement tool is capable of signalling to the app the coordinate of each vertex by measuring the distance from the previous one.

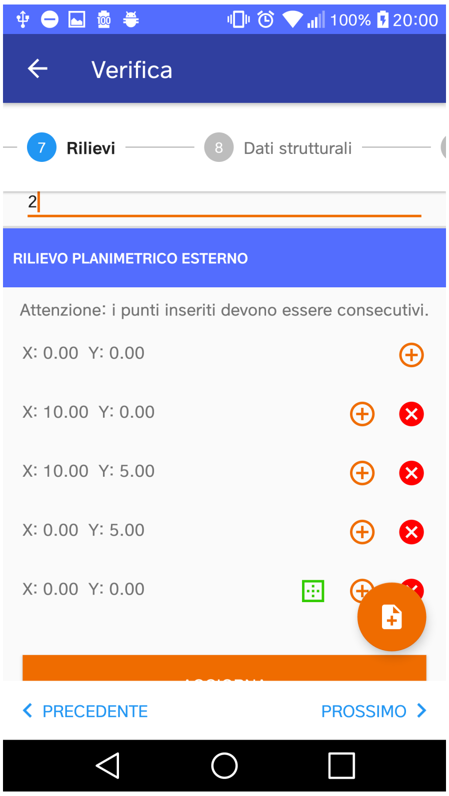



Figure 3.19. Take-over fragment pt.1

To help the user input the vertex three buttons are provided to add, remove or close a point in the graph and each time a new point has to be inserted it takes by default the values of the previous one, since usually the vertex is perpendicular between them, or at least have a coordinate in common. Once the plant has been inputted the button will display it as a graph and calculate its area, perimeter and centre of gravity and will display the latter with a big red dot.

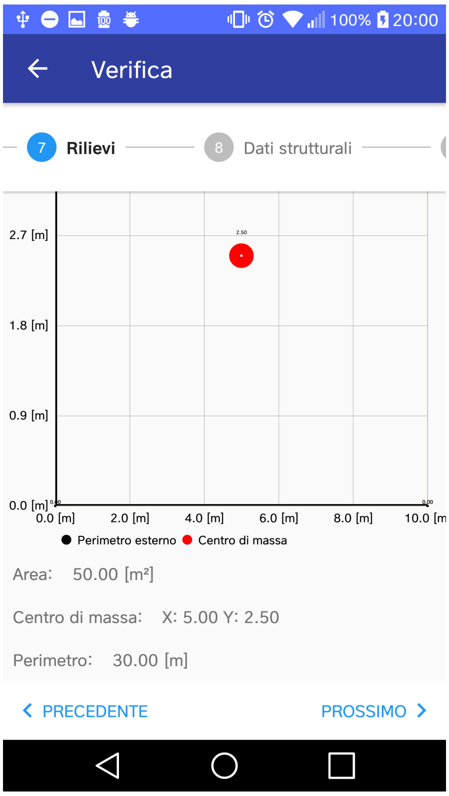



Figure 3.20. Take-over fragment pt.2

The structural data are data regarding the attic and coverage weight. It is a specific weight that will be used to calculate the total weight of the building using the area that was previously inserted in the takeover fragment and that can be seen in a label at the bottom. Both this and the previous fragments are mandatory.


Figure 3.21. Structural data fragment

The column fragments contain all the data regarding the column and its physical composition. At the top the user can choose between different types of concrete and steel, checking their specific data. The real measurements of the column must be inserted, length, width and height together with the data of the steel bars inside of it. Once all these data have been set, the column domain graph will be displayed together with all the points that make it up. As per requriment the program consider all the column identical, even if the influence area is usually different for each pillar (for example the columns on the edge of the building would have a smaller influence area compared to central ones).

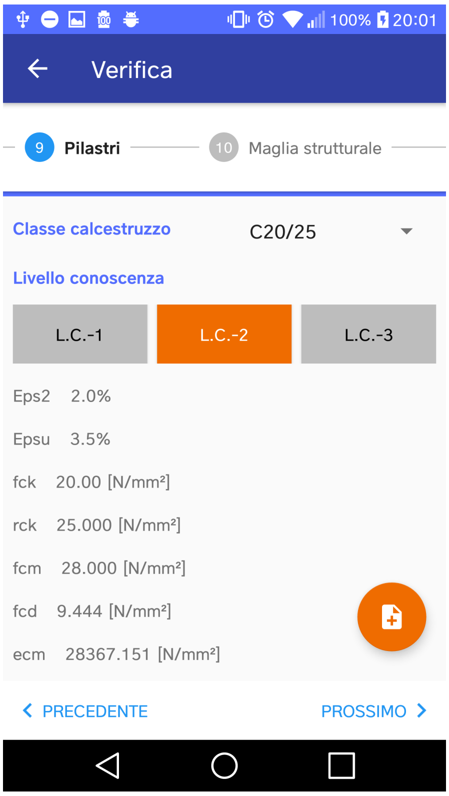



Figure 3.22. Column fragment pt.1

Anyway it is important to remember only the parameters displayed on this fragment will be used to the compute the interaction diagram of a single generic pillar with the selected input data. If the user wants to calculate the vulnerability index for each column it would still be possible by changing the influence area in the next fragment and the physical data of the column accordingly, but he must remember that only one result (the last) will be stored.

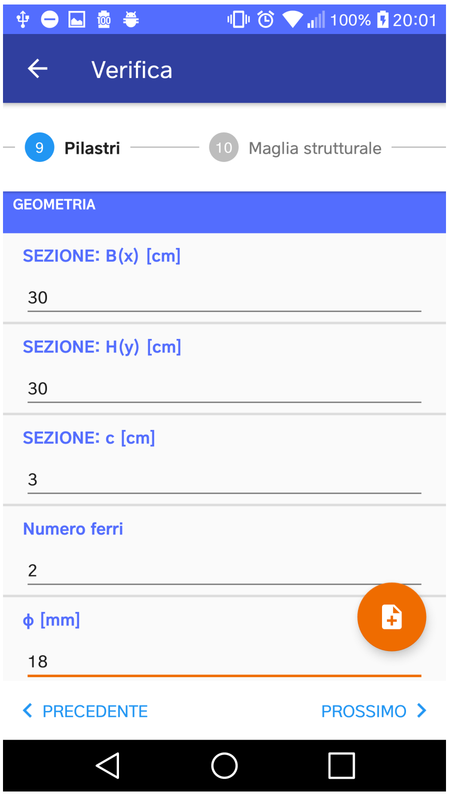



Figure 3.23. Column fragment pt.2

The column domain is the graphic representation of the M-N interaction diagram that the column with specific characteristics can resist. Right under the graph there's a list of point that are used to calculate it, in this way the user can easily check whether the calculation is correct or not.

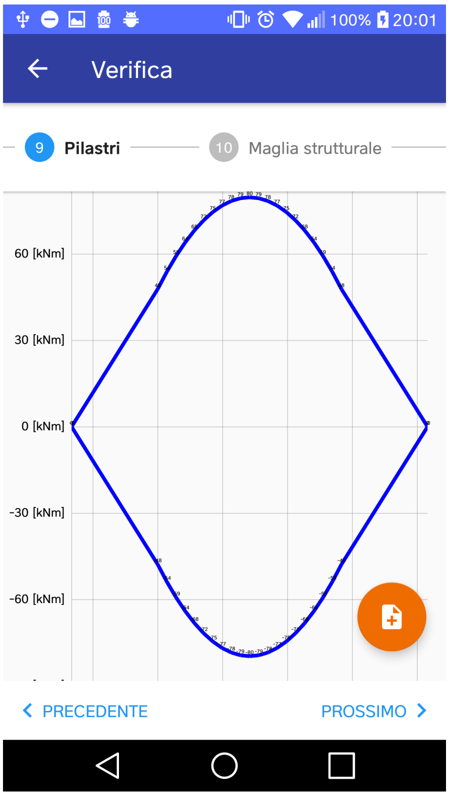



Figure 3.24. Column domain

Once the column domain is set, it is time to input the number of column of the building. For this operation the simpler thing to do is to lay down a grid that cover at least the area of the column and for each point in the grid select if a column is present or not. By default, all pillars are present but, in case of a particular shape of the building like an L, some column can be removed just by touching the correspondent point on the grid. Only the ‘on‘ column, highlighted in green will be counted.

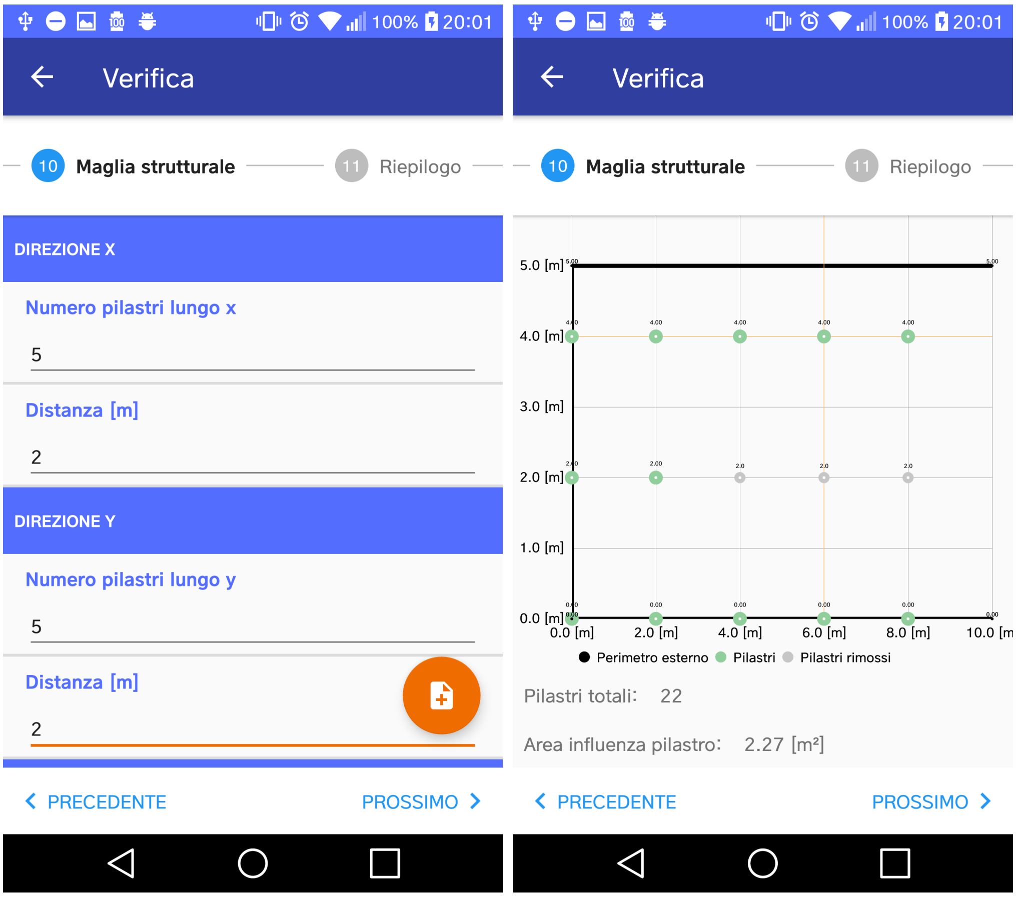



Figure 3.25. Column layout fragment

This fragment contains all the significant data for the calculations, ordered by category, together with a visual representation of the column domain and the loads of each specific state to provide an instantaneous feedback to the user regarding the loads on the column and in general the situation of the building. It is useful to summarize all inputs and to check if something important has a wrong value or different from the truth.

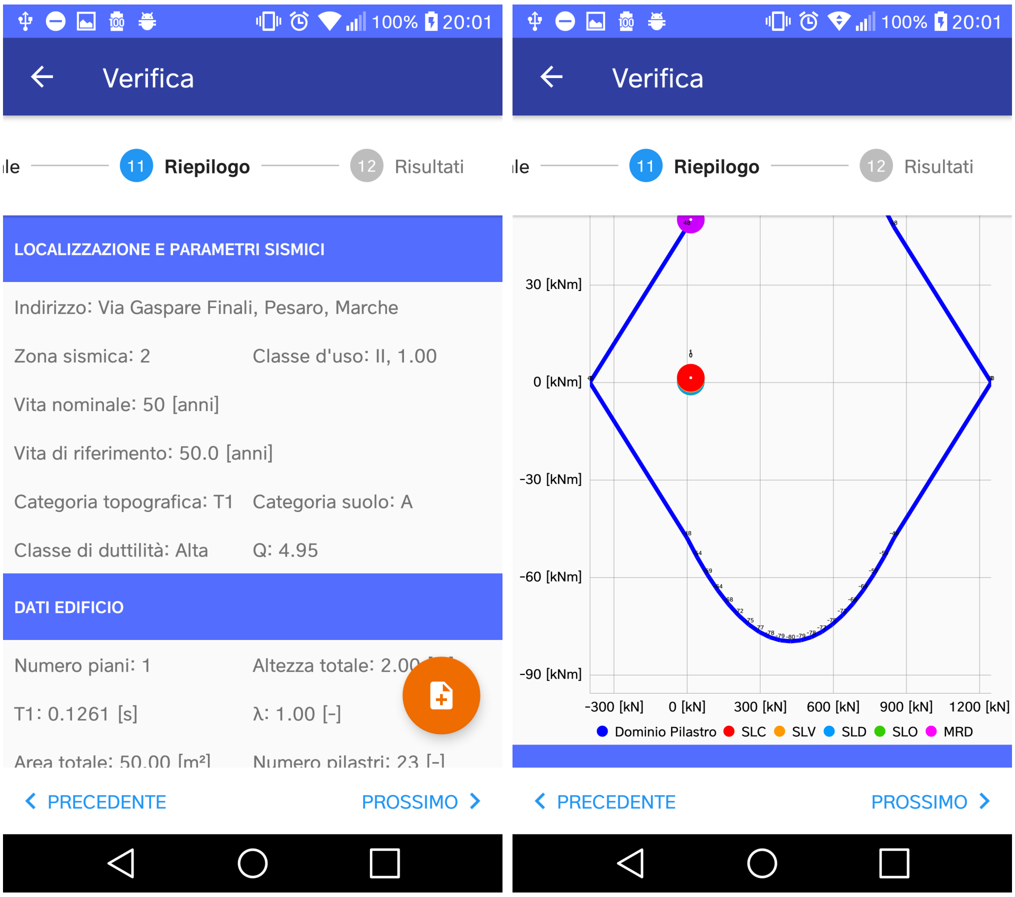



Figure 3.26. Summary fragment

The result fragment shows the calculation of the vulnerability index for each limit state and every intermediate step, using color to highlight danger. The intermediate steps represent the loads and solicitation on each single column, and they represent both the horizontal and vertical momentum that the pillar should resist when a force is applied to it, typically an earthquake. On the left of the coloured bar is displayed the vulnerability coefficient, calculated by dividing Mrd by Msd. On the right of the bar there's the vulnerability index, defined as the vulnerability coefficient percentage, if greater than 100 it is trunkated to 100%.

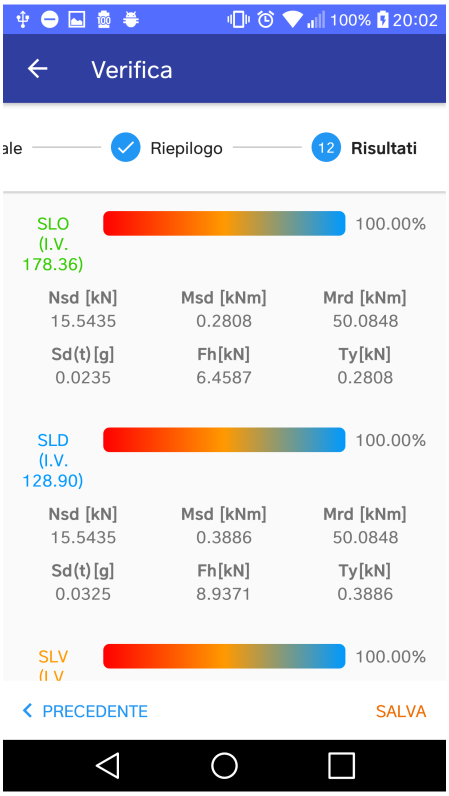



Figure 3.27. Results fragment

The next fragment is not from the report activity but from the main activity that allows the user to check its own data and all the reports he filled to time. Each report has a painted bar that indicates its vulnerability together with some identifying information like the address, when the report was made and the size of the media inside of it. This allow the editing of an already saved report, together with providing the button to export the report in pdf (feature not yet available), to delete the local report from the database or to upload it to the academic server for research purposes as a JSON string.

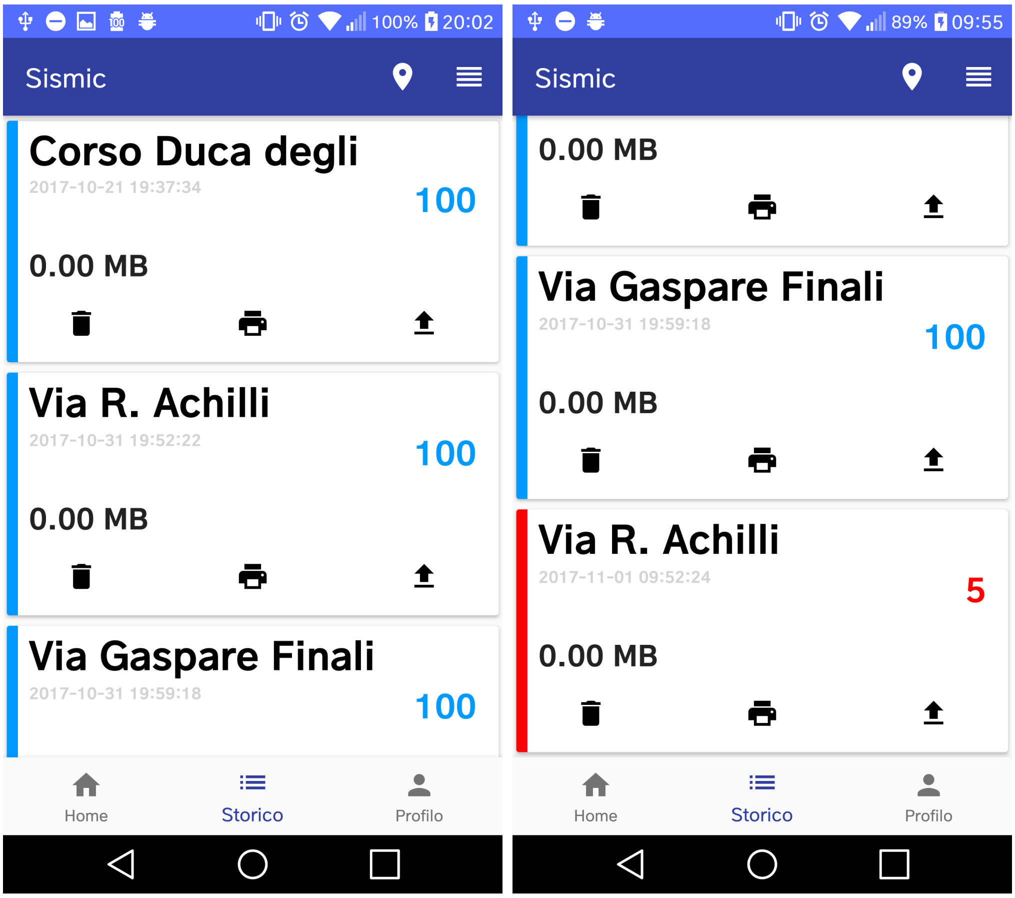



Figure 3.28. Historic fragment

### 3.2.8 Server

Last part of the implementation phase regards the detailed description of the server. Its main job is to provide a solid, persistent and centralized data storage for the reports made through the application, that otherwise would just be available on the local mobile devices where the app is installed. The server is written in Python using the Flask framework for the networking part, together with SQLAlchemy that provides a simple ORM (Object-Relation Mapping) to directly manipulate the database using classes instead of queries. The only specific requirement regarding the server is to be able to receive and store a report made through the application. For this reason, the server is a very simple one.
The modules that compose the server are three: database, model and server. The database module is in charge to handle the database file and to define the database session. The model module allow the definition of the different tables of the database using classes, inheriting from SQLAlchemy Base class and defining primary and foreign keys; the SQL framework will handle all the process of creating or modifying tables. The server module uses Flask framework to include the defined database and

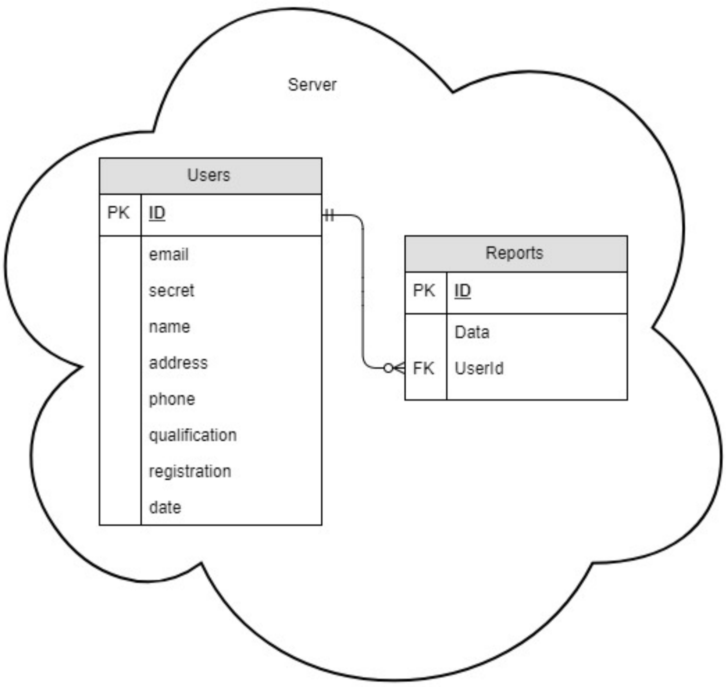



Figure 3.29. E-R Server database

to properly start the server. It also uses @*app.route* decorators to expose specific URL that will trigger specific routines in the database, allowing the interaction with the app through http requests and server-specific result codes. Together with the report data storage, the server allows also a simple registration and login procedure, to keep track of the application users. The security at the moment is not a priority and the only ciphered data is the user's password.
The server provides three access point to the application: registration, login and report upload implemented through three different @*app.route*:

1. @$app.route('/seismic/registration_form', methods = ['POST'])$: the post parameter are the user data: email, password, name, address, phone, qualification and register. This routine check whether the user already exists and if the email and password are valid, then proceeds to store the user ciphering the password and answer success to the application. Otherwise an error occurs and will be notified to the app user.

2. @$app.route('/seismic/login_form', methods = ['POST'])$ The only parameters are email and plain password, the routine checks if the user is present and grant access.

3. @$app.route('/seismic/upload_report', methods = ['POST'])$ Read the JSon report from the Post request and stores it into the Reports table. The only parameter is the user email that surely comes from a logged user.

Other methods make possible to display all the users currently registered and all the reports currently uploaded in JSon string form, allowing just to store and view these data without really using them for anything for the time being. When the census project will become reality, and more specific requirements will be requested, this JSon strings will become classes of the database and will be represented properly into the html page. The server also provides a functionality for storing the media files of the reports, but for now this feature is blocked.
This very simple server module lives onto an Apache server machine in the Software Engineering department of the Politecnico. It has a public available IP that makes it reachable by the devices connected to the internet.

The deployment of the Flask module is accomplished by a *wsgi* Python module that is called through a .conf file very similar to:

```
{
<virtualhost *:80>
ServerName seismic.app
ServerAlias seismic.app.natasha.polito.it
ErrorLog /home/tesi/Seismic/logs/error.log
CustomLog /home/tesi/Seismic/logs/access.log combined
WSGIDaemonProcess seismic user=tesi
WSGIScriptAlias / /home/tesi/Seismic/Server/seismic.wsgi
<directory /home/tesi/Seismic>
Require all granted
WSGIProcessGroup seismic
WSGIApplicationGroup GLOBAL
WSGIScriptReloading On
</directory>
</virtualhost>
}
```

This file tells the wsgi module on which port to run, the name of the server and other configuration data regarding logs, folders and access permissions.

## 3.3 Glossary

1. **User**: subject authorized to view the web portal and the reports.
2. **Technician**: subject that compiles the report on the app and that uploads it to the server.
3. **INGV**: Istituto Nazionale Geologia e Vulcanologia, it provides the databases containing the seismic zone for every Italian municipality and the seismic acceleration parameter for every default return time.
4. **GSP**: Google Service Provider entity used to retrieve position data using Maps and Places API and current position of the device.
5. **CSLP**: Consiglio Superiore Lavori Pubblici: Italian government authority that provides data regarding seismic action activity on the territory. The provided data is .csv with .5 double precision on latitude, longitude and seismogenic data.
6. **WebPortal**: web server capable of receiving a JSON report and storing it into a database. Capable of registering and logging a user.
7. **Media**: picture, video, audio note, sketch, written note that can be saved by the app.
8. **Technical parameters**: site and building parameters useful to calculate seismic hazard.
9. **Seismic database**: database on an academic server that will be used to store report data.
10. **Nominal life**: the nominal life of a structural work is defined as the number of years in which the work, subject to ordinary maintenance, must be able to be used for the intended purpose.
11. **Use class**: is the strategic importance of the building.
12. **Reference life**: reference time period that takes into account nominal life and use class.
13. **Limit State**: performance of building, including structural elements, during an earthquake. Four limit states can be identified SLO (Stato Limite di OperativitÃ ), SLD (Stato Limite Danno), SLV (Stato Limite Vita), SLC (Stato Limite Collasso) Each one these limit states indicates a minimal behaiour that the building should show to resist during an earthquake.

14. **Response spectra**: Elastic response of the acceleration horizontal component, each spectrum is referred to a specific limit state.
15. **Soil category**: local site response evaluation, as indicated in Italian laws.
16. **Topographic category**: local site topographic conditions.

# Chapter 4

# Result

The final result of this thesis work is a functional and efficient Android application that can be used to evaluate the seismic hazard of a building. The result can be considered valid after an extensive phase of testing, performed by Denis Zucchini and the comparison of the application result with the result of today's available technologies such as the pre-mentioned SAP2000 and Spettri-NTC. The validation and result comparison phase can be split in two parts: the first part includes all the calculation and interpolation regarding the seismic action on the Italian territory, such as the identification of the surrounding square nodes and interpolation of parameter that is performed by the Spettri spreadsheet. The second part includes, instead, all the calculation performed for the physical building to determinate the various loads on the column, that is performed by SAP2000.

### 4.0.1 Sap2000

SAP2000 is a proprietary FEM (Finite Element Method) software specifically designed for civil engineering. Is a very versatile tool capable of analysing many and different structures very precisely. It comes with a simple, very CAD/CAM like interface, an analytical and efficient calculation engine and sophisticated verification tool, in this way verification, modelling and analysis are integrated processes of a single environment. It allows also every kind of analysis that a civil engineer may need, from simple two-dimensional examples to real life complex buildings and infrastructures. This program will be used, together with Spettri-NTC to evaluate the accuracy of the application. Clearly the structures that will be analysed with the mobile app will be just a part of all the possible buildings that SAP is able to deeply analyse. In this case the goal is to be 'sufficiently' accurate the be a valid alternative to SAP in few predefined cases where the calculations are simpler (see 3.2.4) but provide stub functionalities for future implementations and scalability.

### 4.0.2 Comparison

The application shows a 100% accuracy, compared to Spettri, in choosing the surrounding square nodes after a consistent set of 50 tries; for this reason, it won't be necessary to compare result from different locations, being sufficiently sure that the nodes picked by the app and Spettri will always be the same, the comparison will take into account just the parameters calculated after the interpolation.
The input data were chosen accordingly to the previous work from Denis Zucchini, in this way it has been easier to evaluate performances between the app, the prototype from the previous thesis work and the official tool provided by the Italian department. These data refer to a building in the Politecnico di Torino area that will be later analysed and compared using SAP2000 too.

Table 4.1. Spettri test data 1

| Parameter | Value |
|---|---|
| Longitude | 7.6622 |
| Latitude | 45.0624 |
| Seismic Zone | 4 |
| Reference Life | 50 |
| Soil Category | A |
| Topographic Category | T1 |
| Ductility Class | CDA |
| Q | 4.95 |
| $\eta = 1/Q$ | 0.202 |

The following pages will show some comparison graphs made using Google Drive functionalities:

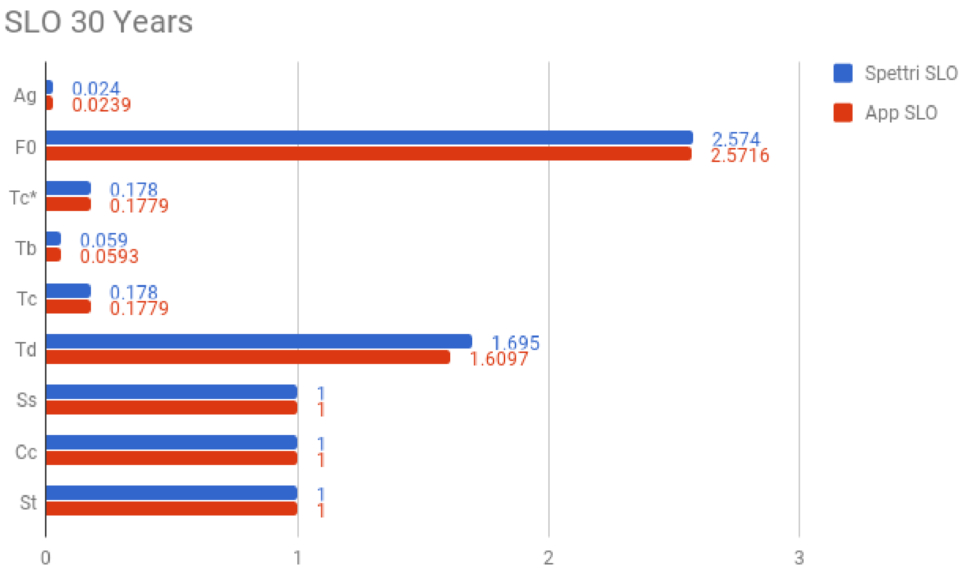



Figure 4.1. SLO Comparison

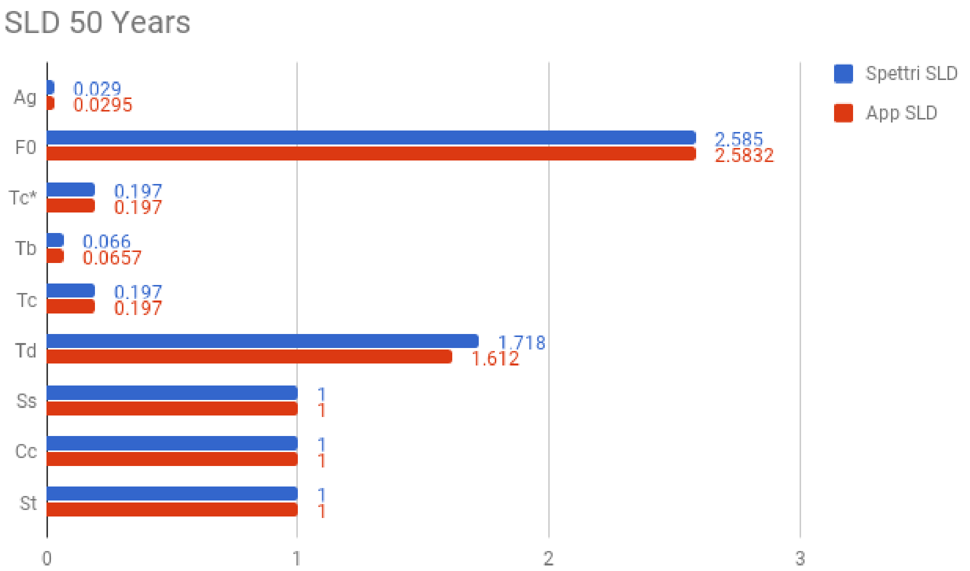



Figure 4.2. SLD Comparison

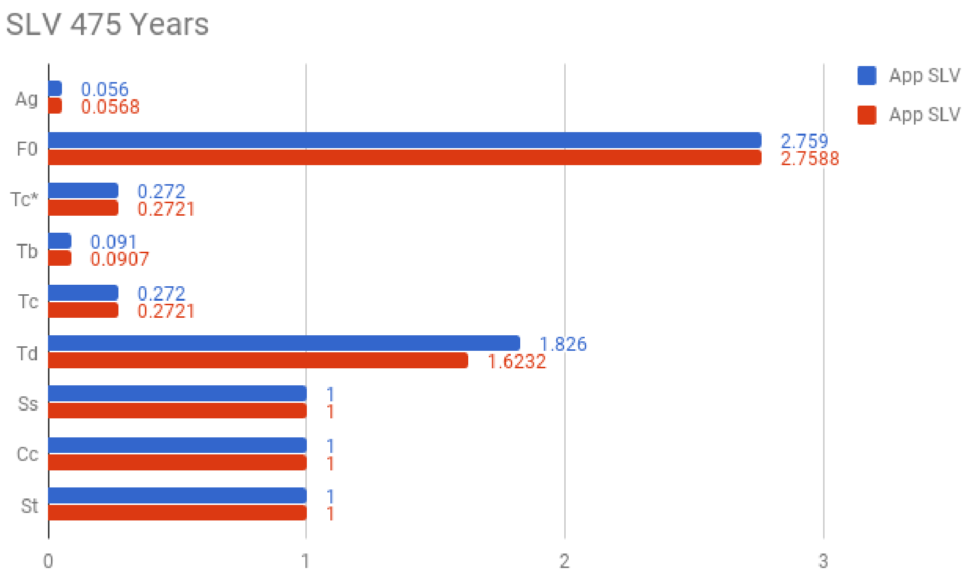



Figure 4.3. SLV Comparison

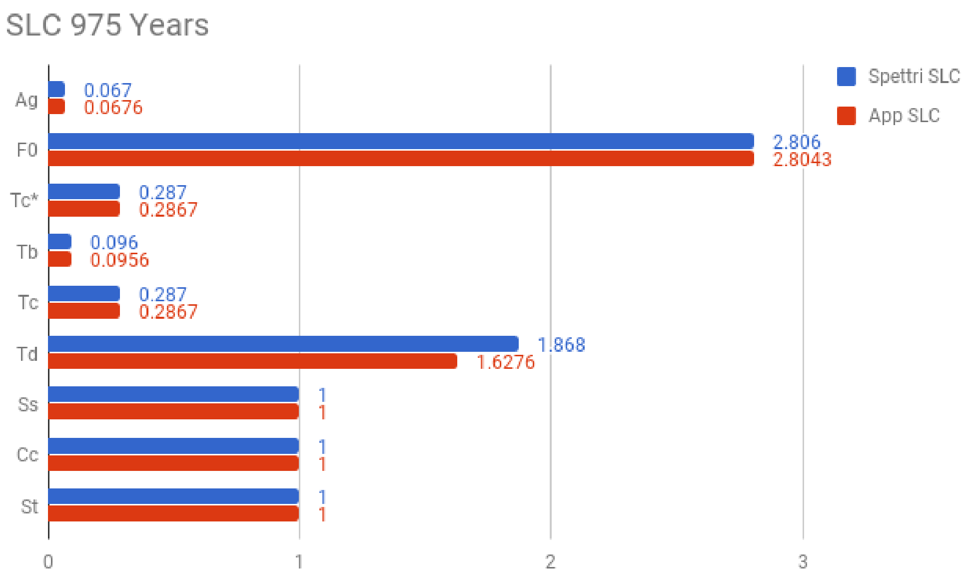



Figure 4.4. SLD Comparison

The graphs show the comparison between Spettri and the App result with the 4.1 reference input data. It is clear that result is more than satisfactory and sometimes even more accurate than Spettri. Since the same functions and calculation are applied to the same parameter B it can be safely assumed that the represented behaviour will be the same in both app and Spettri, with some minimal approximation. An interesting thing to point out is that modifying the initial input data with the following, doesn't change the parameter at all since the Q value intervenes only during the acceleration function calculation and doesn't change the other parameters .

Table 4.2. Spettri test data 2

| Parameter | Value |
|---|---|
| Ductility Class | CDB |
| Q | 3.30 |
| $\eta = 1/Q$ | 0.303 |

Second set of input data: input regarding physical properties of the building itself.

Table 4.3. Building test data

| Parameter | Value |
|---|---|
| Floor number | 1 |
| First floor heigth | 3[m] |
| Total height | 3[m] |
| External perimeter vertices | 0.0 - 10.0 - 10.5 - 5.0 - 0.0 |
| Area | 50 |
| Perimeter | 30[m] |
| Mass center | 5 - 2.5 |
| T1 | 0.171 |
| G1 | 2.35 |
| G2 | 2.00 |
| QK | 2.00 |
| QTot | 5.45 |
| Building weigth | 272.5 |
| Concrete class | C25/30 |
| Concrete knowledge | L.C.-3 |
| Steel class | B450C |
| Steel knowledge | L.C.-3 |
| B(x) | 30 |
| H(y) | 30 |
| Concrete cover | 3 |
| Column number | 2 |
| Sled bar diameter | 18 |
| Column number | 6 |
| Influence column area | 8.33 |

The second phase, and in general the whole application performance, has been tested against SAP2000. In the next page will be described the test data used by both SAP and the application for the description of the building. For the whole process of evaluation of building both set of data (the first set is 4.1) must be inserted into the app and the next table shows the complementary data set that will complete the whole set of input data for the Android application. An important thing to point out is that the majority of the following data is calculated automatically by the app while with SAP the user must input all of them one by one.

The first comparison of the second phase will take into consideration T1 parameter, to check whether the calculation of the horizontal component of the acceleration for this building is correct. This comparison is yet again performed against Spettri and again the difference is minimal, only due to the different accuracy of the two instruments

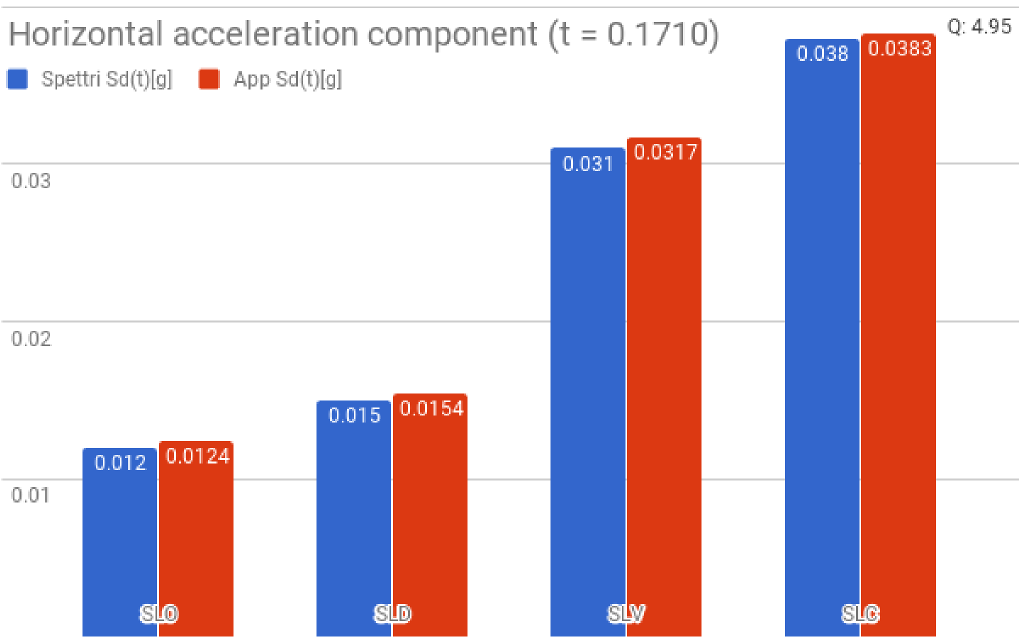



Figure 4.5. Horizontal component comparison 1

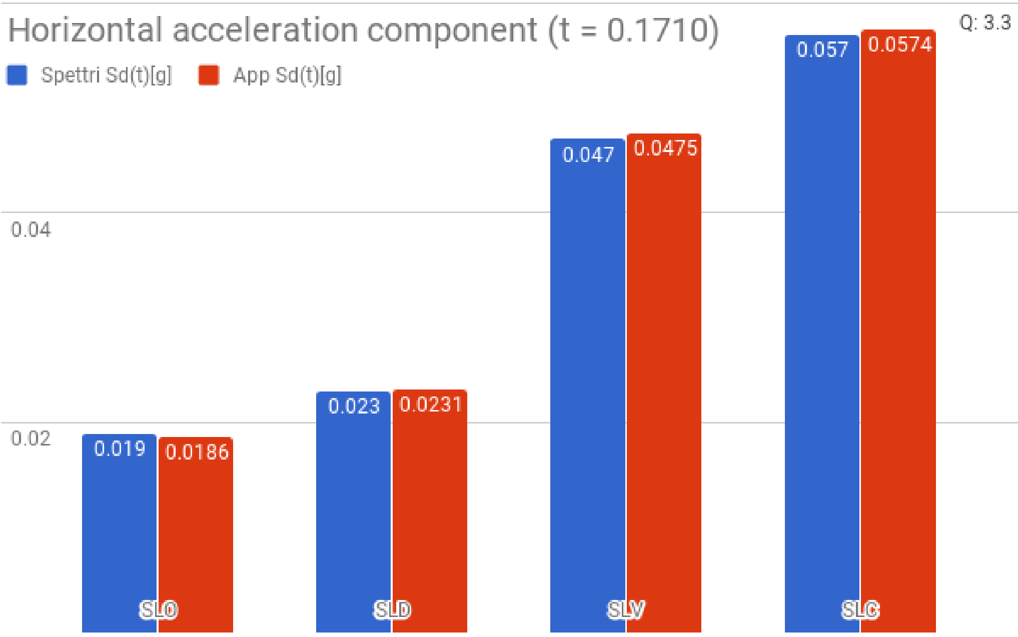



Figure 4.6. Horizontal component comparison 2

The second comparison of this phase will take into account the column domain and will take into account the loads of each limit state on the single column, together with considering the Zucchini prototype as well. This proves that the Android tool is valid alternative to Spettri as it can provide a quick and accurate estimate of the limit state parameter function, showing just the interesting data in a more fancy way. These comparison also confirm again that the method described in Zucchini work is accurate enough to provide a powerful asset to technician for the seismic hazard assessments process.

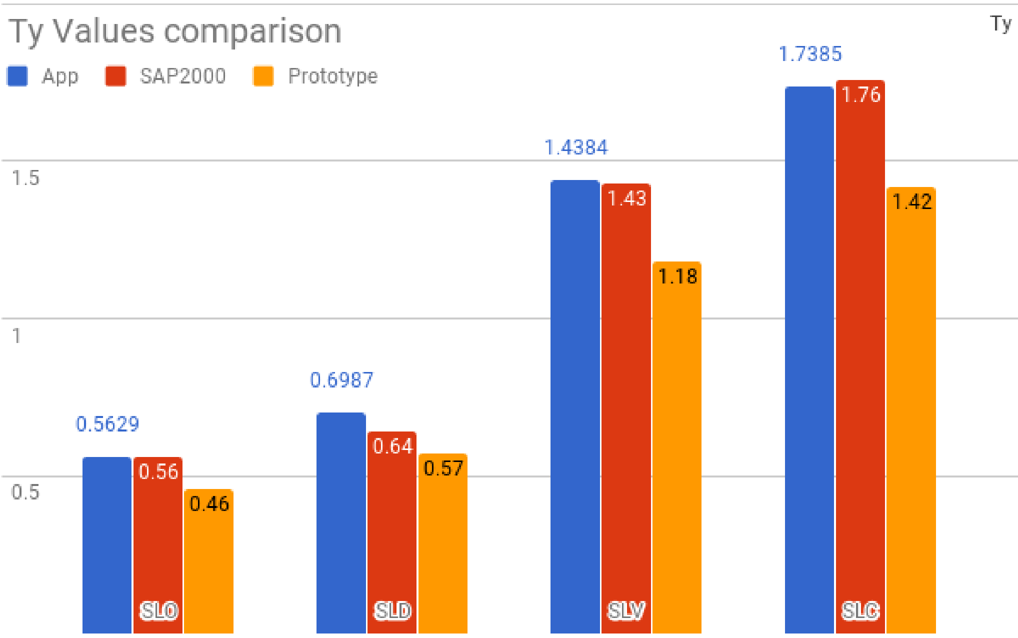



Figure 4.7. Ty: loads on the single column

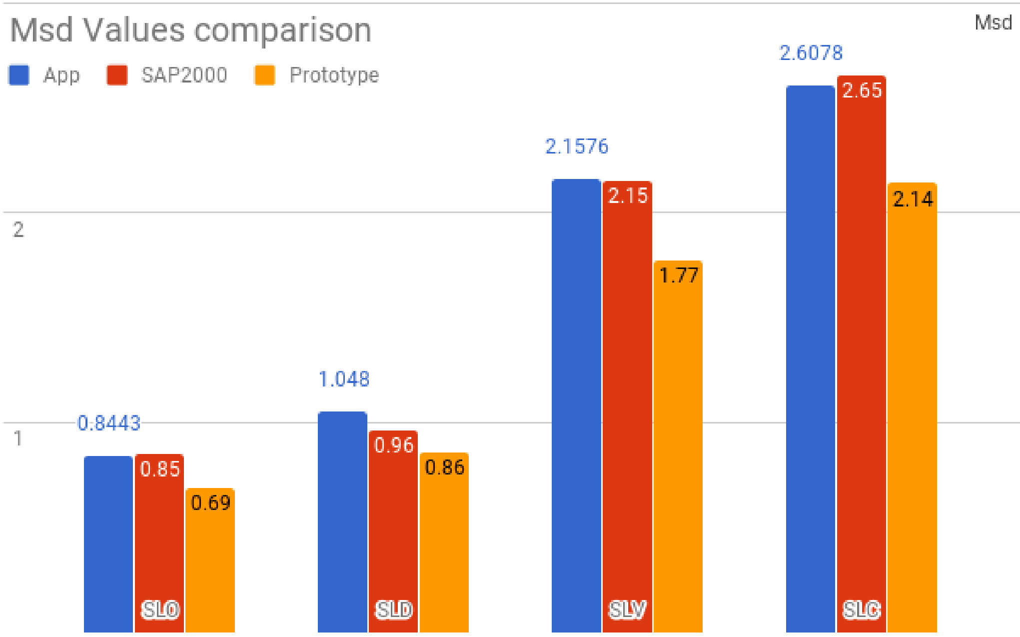



Figure 4.8. Msd solicitation

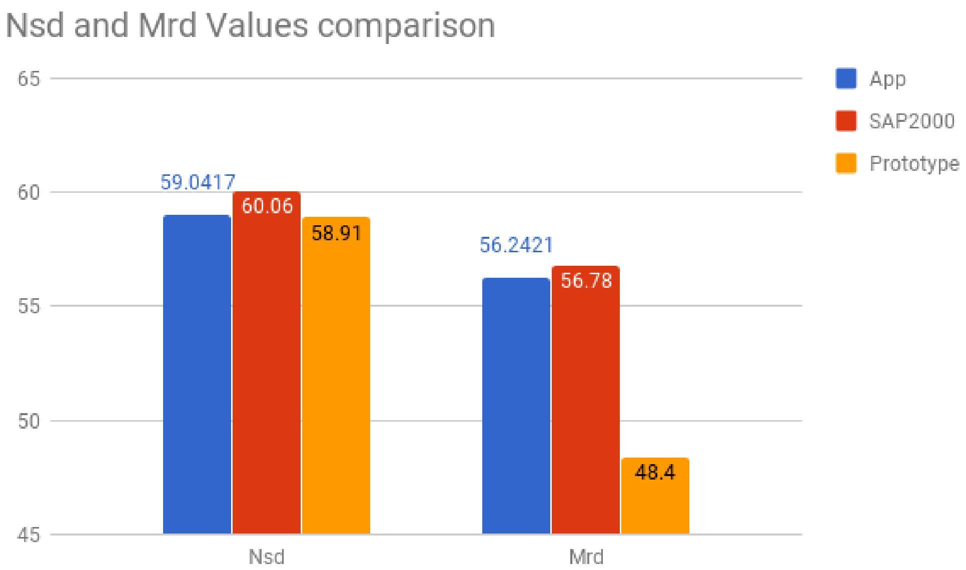



Figure 4.9. Mrd and Nsd solicitation are the same for every limit state

Last comparison is between the vulnerability coefficient results for each limit state, considering the previous results. This coefficient will be transformed into a percentage indicating how safe is the building, by multiplying the coefficient for 100. If the percentage is greater than 100% this last value will be considered, meaning that for the specific limit state, there's no risk that the building could fail the expectations. Every other value range will have a different meaning and will be identified with a specific color indicating danger. A percentage between 0 and 50% indicates a sure hazard for the limit state requirements specification.

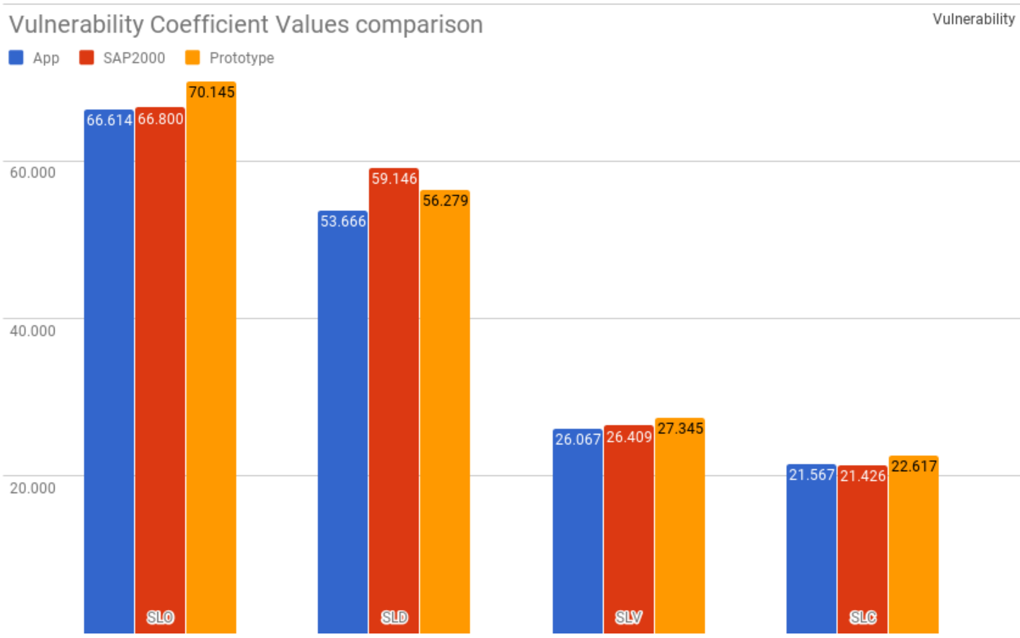



Figure 4.10. Vulnerability coefficient

# Chapter 5

# Conclusion

This thesis work describes in detail all the process of software engineering of a very technical mobile application. It shows that even complex calculation and methodologies can be accurately simplified and efficiently implemented by using Kotlin for Android, the best mobile development tool available at the moment.
The work can be summarized in three main steps. The first step is about gathering information and collect functional, non-functional and domain requirements, comparing the current available tools for the job and identifying their weak spot and strength to improve the first and empower the latter. The second step consisted into using the cleaner architecture for software engineering to correctly implement a scalable and usable software that together with writing understandable, crystalline and commented code make this application a very useful tool for academic studies. The last step was about verifying if the work was actually usable and accurate a any other seismic hazard vulnerability calculator, to make it a feasible solution for technician that want to speed up takeovers and all the process of seismic vulnerability assessment. This tool makes possible the objective stated in the first paragraph: it can be used to fill a national census of buildings.
Moreover, as a personal note, I would like to add that when a statistically valid number of report will be present in the database it will be possible, using a neural network, to analyse all the possible correlation between the stored data to study if some correlation between a specific parameter and the seismic vulnerability exists. This could be a failure but could also lead to a better understanding of which simple parameter are an accurate indicator of seismic hazard, without the need of all other more complex data. The simple architecture and frameworks used to implement the Android application and the server make also this project very suitable to further development to make it more usable by, for example, including more complex buildings or providing a user interface to the web server. In general in the future it will be possible to remove all the semplification choosen for this project, allowing in this way a more generic analysis of buildings.

# Appendix A

# Closest nodes algorithm

The application requires the calculation of seismic data regarding the probable terrain acceleration of particular site during an earthquake. These data will be used to calculate the generic horizontal force applied to the building and to each single column during the earthquake. For the purpose the Italian CSLP provides a .csv file, ordered by ascending longitudes, containing acceleration data of particular points on the Italian soil for default return times, these data are represented by three parameters (ag, and F0, tc*) that are used to describe the acceleration function. It is very unlikely that a building is exactly over one of the entries of the database, so some interpolation is needed to adjust the data for each the building.

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | OBJECTID,"ID","LON","LAT","T30ag","T30F0","T30Tc","T50ag","T50F0","T50Tc","T72ag","T72F0","T72Tc","T101ag","T101F0","T101Tc","T140ag","T140F0","T140Tc","T | | | | | | | | | | | | | | |
| 2 | 1,13111,6.544813,45.13446,0.26297,2.4951,0.17895,0.34013,2.5099,0.20544,0.3944,2.5455,0.21943,0.46949,2.4898,0.23635,0.54488,2.5002,0.23866,0.63955,2.494 | | | | | | | | | | | | | | |
| 3 | 2,13333,6.550592,45.08463,0.2641,2.4941,0.17886,0.34067,2.5116,0.20522,0.3945,2.5482,0.21931,0.46882,2.4939,0.23618,0.54335,2.5042,0.23872,0.6361,2.5005,0 | | | | | | | | | | | | | | |
| 4 | 3,13555,6.556428,45.03479,0.26424,2.4957,0.17863,0.34007,2.5146,0.20479,0.3934,2.5514,0.21907,0.46624,2.4998,0.23598,0.53991,2.5087,0.23892,0.62998,2.508 | | | | | | | | | | | | | | |
| 5 | 4,13777,6.562108,44.98496,0.26344,2.4998,0.17825,0.33844,2.5179,0.20423,0.3912,2.5549,0.2187,0.46197,2.5073,0.23586,0.53477,2.5136,0.2393,0.62149,2.519,0. | | | | | | | | | | | | | | |
| 6 | 5,12890,6.609601,45.18835,0.28393,2.4598,0.19386,0.36432,2.507,0.20974,0.4309,2.501,0.22097,0.50942,2.4803,0.23581,0.5852,2.4997,0.24093,0.69462,2.4688,0. | | | | | | | | | | | | | | |
| 7 | 6,13112,6.615325,45.13852,0.28631,2.4574,0.19394,0.36635,2.5082,0.20971,0.4334,2.5009,0.22082,0.51126,2.4843,0.23552,0.58641,2.5038,0.24574,0.69542,2.472 | | | | | | | | | | | | | | |
| 8 | 7,13334,6.621044,45.08868,0.28767,2.457,0.19382,0.36716,2.5108,0.20957,0.434,2.5032,0.2208,0.51124,2.4879,0.23543,0.58578,2.5072,0.24587,0.69384,2.4766,0. | | | | | | | | | | | | | | |
| 9 | 8,13556,6.626787,45.03884,0.2881,2.4597,0.19346,0.36692,2.5144,0.20934,0.433,2.5079,0.22083,0.50973,2.4915,0.23548,0.58378,2.5103,0.24598,0.69059,2.4808,0 | | | | | | | | | | | | | | |
| 10 | 9,13778,6.632496,44.98901,0.28751,2.4639,0.19294,0.36561,2.5183,0.20909,0.4304,2.5149,0.22085,0.50675,2.495,0.23563,0.58041,2.5133,0.24602,0.68561,2.4855, | | | | | | | | | | | | | | |
| 11 | 10,14000,6.638289,44.93916,0.28598,2.4689,0.19234,0.36334,2.522,0.20872,0.4264,2.5236,0.22079,0.50241,2.4983,0.23588,0.57572,2.5159,0.24133,0.67884,2.490 | | | | | | | | | | | | | | |
| 12 | 11,14222,6.64387,44.88933,0.28377,2.4744,0.19176,0.36041,2.5251,0.20817,0.4214,2.5332,0.2207,0.49708,2.5013,0.2362,0.57009,2.5181,0.24141,0.67074,2.4964,0 | | | | | | | | | | | | | | |
| 13 | 12,12891,6.680284,45.19237,0.30591,2.4299,0.1987,0.38896,2.5047,0.21235,0.467,2.469,0.23079,0.54433,2.4877,0.23481,0.62544,2.4969,0.24727,0.73616,2.4732,0 | | | | | | | | | | | | | | |
| 14 | 13,10228,6.682573,45.79445,0.28338,2.4188,0.19749,0.36433,2.4631,0.21637,0.4299,2.4609,0.23867,0.50495,2.441,0.2467,0.57737,2.4775,0.25737,0.67935,2.4824, | | | | | | | | | | | | | | |
| 15 | 14,13113,6.685977,45.14252,0.30854,2.4283,0.19897,0.39123,2.5078,0.21241,0.4695,2.4719,0.23071,0.54625,2.4922,0.23465,0.6272,2.501,0.24731,0.73711,2.4781, | | | | | | | | | | | | | | |

Figure A.1. Database provided by the CSLP

The algorithm takes in input current latitude and longitude and must return something really like Spettri-NTC 2.6: four surrounding pair of coordinate each representing a row in the database.

**Algorithm 1** Entry point

**procedure** FINDSQUARENODESOF<br>
    *result* ← mutablelist<br>
    *startIndex* ← *insertInOrder longitude*<br>
    **if** *startIndex* >= *size-1* **then return** result<br>
    **if** *startIndex* < 0 **then return** result<br>
    *leftRes* ← *innerGetClosestPoint* ← *params* :<br>
    *rightRes* ← *innerGetClosestPoint* ← *params* :<br>
    *result* ← rightRes.add.leftRes<br>
    *returncreateSquare* ← result

The input parameter for the following function are: x: Double, y: Double, left: Boolean, startIndex: Int, limitDistance: Double, tmpList: MutableList<Triple<String, Double, Int». X and Y indicates the target coordinates, left is the direction, startIndex is the index in the nodes array from which the search should start, limitDistance is the greater distance calculated until now, tmpList is the temporary result list of nodes, indicated by their id, distance from target coordinates and index.

**Algorithm 2** Recursion

**procedure** INNERFINDSQUARENODESOF<br>
    *result* ← mutablelist()<br>
    *nextNode* ← Nodes[startIndex]<br>
    *newdistance* ← *calculateDistance(x,y, nextNode.x, nextNode.y)*<br>
    **if** *newdistance* > limitDistance **then return** tmpList<br>
    *newIndex* ← tmpList.addAll(*getCloseNodesTo(startIndex, left)*<br>
    *tmpList.orderByDistance*()<br>
    *maxDist* = *tmpList.sumDist*()<br>
    *InnerFindSquareNodesOf(x,y, left, newIndex, maxDist, tmpList).*

This algorithm converges thanks to the stopping condition at line. In fact, since the algorithm starts from the closest longitude point and moves into the two directions if the distance between the target coordinates and the next node to analyse is greater than the sum of the candidate node square, it is impossible that the next nodes will be closer and they will never become vertex of the candidate square.
The ‘createSquare‘ method simply iterates through the result trying to put each

point into the square and calculates the distance. If a correct, four square vertex figures is created the result will be considered valid.
The algorithm implements an ILS (Iterated Local Search) exploiting the property of the database being ordered on longitude. Starting from the closes longitude to the current the algorithm branches two ways recursively until the stopping condition is satisfied. Once the condition is met the two branches will merge the result by eliminating the farthest entries, and keeping only the four that satisfy the condition to form a square around the current point. The stopping condition states that if none of the current local nodes list have a distance is lesser or equal to the sum of the distances of the current probable square the algorithm will stop. Being ordered only on longitude may have the problem to start the search from a very low point (Sicily for example) while the result is on a similar longitude but higher on the map (like Friuli). If the search starts from a very far point a branch will die immediately while the other will iterates to the closest result. Some optimization implemented include storing the current minimum node distance, only recalculating when the candidate square changes and the reading from the database file is performed only once.

# Appendix B

# Seismic spectra calculations

Once the closest nodes have been identified, the app will have to interpolate the data from the four surrounding node and interpolating them to retrieve the current parameters and build the acceleration response function for the current site. The interpolation is a simple inverse-distance base interpolation where each parameter (ag, F0, Tc*) can be represented by:

$$p = (\sum_{i=1}^{4} p_i / \sum_{i=1}^{4} d_i)/(1/\sum_{i=1}^{4} d_i)$$

where p is the single parameter (ag, F0, Tc*) for each return time in the database.

With the correct acceleration parameters now, the program will use the reference life set by the user to retrieve the right column in the database to use. Until now it considered all the limit states, but when the reference life is known the program will show only the acceleration functions of the corresponding limit states. Usually the year of the limit states match with the one that can be read from the database but when the year don,t match, another interpolation, this time logarithmic must be performed to use the correct data to build the function:

$$log(p) = log(p_i) + log\left(\frac{p_2}{p_1}\right) log\left(\frac{T_r}{T_r1}\right)\left[log\left(\frac{T_r2}{T_r1}\right)\right]^{-1}$$

Once the right parameter for the limit states are known the calculation of the function can be performed. The horizontal axis identifies the first seconds of the earthquake and go from zero to 4, the vertical axis contains the acceleration that is usually high at the beginning and low at the end. Again, the mathematical functions used by the app are the same of Spettri-NTC to remain very faithful to the current process.

The four time intervals define the trend of the response function for a specific limit state. The algorithm will then divide the zero to fourth interval in small steps

**Espressioni dei parametri dipendenti**

$S = S_S \cdot S_T$ (NTC-08 Eq. 3.2.5)

$\eta = \sqrt{10/(5+\xi)} \geq 0,55;\ \eta = 1/q$ (NTC-08 Eq. 3.2.6; §. 3.2.3.5)

$T_B = T_C/3$ (NTC-07 Eq. 3.2.8)

$T_C = C_C \cdot T_C^*$ (NTC-07 Eq. 3.2.7)

$T_D = 4,0 \cdot a_g/g + 1,6$ (NTC-07 Eq. 3.2.9)

**Espressioni dello spettro di risposta** (NTC-08 Eq. 3.2.4)

$$0 \leq T < T_B \qquad S_e(T) = a_g \cdot S \cdot \eta \cdot F_o \cdot \left[\frac{T}{T_B} + \frac{1}{\eta \cdot F_o}\left(1 - \frac{T}{T_B}\right)\right]$$

$$T_B \leq T < T_C \qquad S_e(T) = a_g \cdot S \cdot \eta \cdot F_o$$

$$T_C \leq T < T_D \qquad S_e(T) = a_g \cdot S \cdot \eta \cdot F_o \cdot \left(\frac{T_C}{T}\right)$$

$$T_D \leq T \qquad S_e(T) = a_g \cdot S \cdot \eta \cdot F_o \cdot \left(\frac{T_C T_D}{T^2}\right)$$

Lo spettro di progetto $S_d(T)$ per le verifiche agli Stati Limite Ultimi è

Figure B.1. Database provided by the CSLP

and calculate for each fraction of second the relative value on the vertical axis. The first interval, from zero to Tb is a line so the algorithm will calculate only the first and last point. The second interval is constant, and the acceleration keeps the same value it has in Tb until Tc, so no computation here are needed. Last two intervals have a parabolic trend so the remaining time (from Tc to four seconds) will be split in forty steps and for each of them will be calculated the correspondent acceleration. Some optimization can be made since the value of the constant term does not depend on time, so it will be precomputed and passed as parameter to the function calculating the acceleration.

# Appendix C

# Seismic action on building calculations

The calculation regarding the subject building can be split in two different parts. In the first one the app will calculate the generic physical properties of the building: total weight, area, gravity centre and total height. These properties are not easy to be calculated per se but the technician during a takeover has to physically measure the plant and the height of each single floor and has to collect the building documents that describe the materials used to build it. Once the requested parameter is input into the app it will show the intermediate results for the building and will notify the user if the subject building is suitable for the assessment. In facts, for the procedure to work, the building must satisfy specific requirements to allow the static linear analysis:

1. Building material must be concrete.
2. Build maximum height forty meters.
3. Plant and height regular.

Moreover, the domain requirements states that we will consider all the columns identical in physical properties and loads, in this way it won't be necessary to repeat the calculation for each column of the building. This may bring to not very accurate result but it allows to calculate the worst case scenario for a column (the case in which the load on is the greatest) and it is always possible to calculate the interaction diagram for a specific different column by temporary changing the input data.

If these conditions are met, the application will allow the visualization of final part of the report. It contains data regarding the most important part of the building: the columns. The user will be able to input the concrete type and steel used to build the column, together with its shape and composition. These data will allow the calculation of the column domain that represents how much stress the column can handle. The vertical axis of the domain column graph represents the compression (N) force, while the horizontal axis the bending moment (M) stress. Also, here the function is divided in parts: the first and last parts are linear and start from M equal zero, allowing an easy calculation. The middle part has a parabolic trend and depends on the height of the column, N and M can be describe by the two functions:

$$N = Fcd \times b \times h$$

$$M = A_s \times Fyd \times (h - 2d,) + (n \times ((H/2) - (h/2)))$$

Where Fcd and Fyd are specific material parameters, $A_s$ is the total area of the steel in the column armour, h is fraction of the total height of the column (this is the variable parameter), $H_tot$ is the total column height and b and d, are width and distance from armour. Last step is to mirror the function on M positive onto negative M.
Once the domain column has been calculated the application must check if the limit states and their correspondent acceleration can harm the building so for each limit state a point will be calculated in the column domain graph using these two formulas:

$$t_1 = 0.0075 \times H^{3/4} \; N_l = P_a \times SP_v \times 1.3$$

$$M_l = \left(\frac{S_d(t_1) \times T_w \times \lambda}{P_c}\right) \times \left(H/2\right)$$

Where $P_a$ is the column influence area, $SP_v$ is the specific weight of the building and 1.3 is a 30% factor for the column itself, $S_d(t)$ is the acceleration of the limit state read from graph at $t_1$, $T_w$ is the total weight of the building and $P_c$ is the total column count. H is the total height of the building.
This function can be splitted in three different part: the first has a linear trend and starts from

# Bibliography


[1] Denis Zucchini [2017], *'Studio di un'applicazione mobile per il calcolo della vulnerabilitÃ  seismica di edifici esistenti'*

[2] CNR-GNDT, *'Seismic risk for public buildings'*, Part I, Methodological aspects, GruppoNazionale per la Difesa dai Terremoti, Roma.

[3] Dolce M., Moroni C. [2005], *La valutazione della vulnerabilitÃ  e del rischio seismico degli edifici pubblici mediante le procedure VC (vulnerabilitÃ  c.a.) e VM (vulnerabilitÃ  muratura)*, Atti di Dipartimento, Vol. 4, UniversitÃ  degli studi della Basilicata.

[4] GNDT , *Manuale per la compilazione della Scheda GNDT/CNR di II livello*

[5] Dolce, Zuccaro: , *Atti Nuovo Cimento Modello di caratterizzazione tipologica a scala nazionale*, GNDT

[6] Zuccaro Giulio: Progetto SAVE, *SAVE Strumenti Aggiornati per la VulnerabilitÃ  Seismica del patrimonio Edilizio e dei Sistemi Urbani*, SAVE Strumenti Aggiornati per la VulnerabilitÃ  Seismica del patrimonio Edilizio e dei Sistemi Urbani

[7] Mantovani E., Viti M., Babbucci D., Cenni N., Tamburelli C., Vannucchi A. [2012], *A middle term prediction of earthquake in Italy: some remarks on empirical and deterministic approaches*, Boll.Geof.Teor.Appl., 53, 89-111.

[8] Alistair Cockburn [2000], *Writing Effective Use Cases.*

[9] Clean Architecture [Online-Last Accessed 29/10/2017], *https://8thlight.com/blog/uncle-bob/2012/08/13/the-clean-architecture.html.*

[10] Kotlin [Online-Last Accessed 29/10/2017], *https://kotlinlang.org/docs/reference.*

[11] Material Design Guidelines [Online-Last Accessed 29/10/2017], *https://material.io/guidelines/.*

[12] Seismic hazard, Protezione Civile [Online-Last Accessed 28/11/2017], $http://www.protezionecivile.gov.it/jcms/it/rischio_sismico.wp$.

[13] G.P. Cimellaro, G. Scura, C.S. Renschler, A.M. Reinhorn and H.U. Kim, *Rapid building damage assessment system using mobile phone technology.*

[14] Spettri-NTC [Online-Last Accessed 28/1/2017], $http://cslp.mit.gov.it/index.php?option = com_content\&task = view\&id = 75\&Itemid = 20$.

[15] NTC 2008, *Norme Tecniche per le costruzioni DM 14/01/2008.*