APPUNTI PER STILARE IL RASD

During the developing of this project we followed an evolutionary and agile model (to forecast, for example, a service to track movement in general.

**DEFINITIONS**

* Data Anonymization (when possible): deleting the fiscal code associated to every data tuple obtained by the query. If the total number of fiscal codes is less than 1000, refusing to return data.

[è NECESSARIO DARE QUESTA DEFINIZIONE???]

**ACTORS**

**[each use case must lead to one or more requirements]**

**NB:** stiamo assumendo che l’applicazione lavori comunque con un dispositivo indossabile!!

**Altre ipotesi che potrebbe essere necessario inserire**

* Le persone sono identificate tramite codice fiscale
* Abbiamo deciso di considerare anche le situazioni in cui cade la connessione internet etc (che invece potevamo evitarci facendo delle ipotesi) [ho scritto questo a partire da una domanda che ho trovato su beep]

**Schema fisico (DESIGN REQUIREMENT)**

SENSORE ||||| APPLICAZIONE |||||| RETE ||||| SERVER |||| DATABASE

||||| HARDWARE SU CUI GIRA L’APP ||||||

(Nei casi in cui il device (qui sopra definito come *hardware su cui gira l’applicazione*) non sia uno smartwatch (che integra il sensore al proprio interno e ha uan scheda sim per comunicare sulla rete), bisogna includere nello schema fisico anche la rete tra sensore ( che si troverà quindi su di un qualsiasi dispositivo indossabile) e il device ( che sarà quindi con ogni probabilità lo smartphone). Tale connessione sarà di tipo Bluetooth (??).

Abbiamo stimato i dati da trasmettere tra sensore e device in circa 1,2 mb al giorno: la ritengo una cifra accettabile per una connessione bluetooth, che mi sembra la più indicata per trasmettere dati a così corto raggio.

**Approccio prudenziale**

Non possiamo ignorare eventuali malfunzionamenti rilevati nel nostro servizio, ma non possiamo nemmeno mandare un’ambulanza a casa delle persone soltanto perché si è spento un componente.

(trade off) -> In caso di malfunzionamenti, chiamiamo un numero di emergenza.

[ENGLISH ONLY]

**GOALS**

In this part we expose functional requirements and domain assumptions associated to the goals of our application that we want to satisfy, distinguing the two actors.

(legenda: (R): functional requirement, (D) domain assumption, (NR) non-functional requirement

**[ I PROPOSE TO DELETE THE FOLLOWING GOAL]**

1. Provide the user notifications and advice about his health status
   1. ……

[N: B. Ci eravamo chiesti se dovessimo rendere possibile alle third parties di specificare preferenze o dovessero fare ogni volta una nuova query per chiedere gli stessi dati aggiornati. Rileggendo la traccia ho trovato: “*also, it allows the third party to subscribe to new data and receive them as soon as they are produced”.]*

1. Allow third parties to subscribe to new data
   1. The user
   2. When more than 15% of data in a “favourite query” specified by the user changes, the machine sends a notification to the user via e-mail which includes new data(R)

**RIFLESSIONI PERSONALI**

1. Ho cercato di mettere il più possibile i goal in termini di shared phenomena, cioè di fenomeni che possono essere controllati dal sofware (altrimenti che goals sarebbero se non potessero essere garantiti interamente dall’applicazione?), poi ho letto le slide che fanno:

*Goals are prescriptive assertions formulated in terms of world phenomena (****not necessarily shared)***

**EH MA SE NON SONO SHARED COME FACCIO??**

1. *Requirements are prescriptive assertions formulated in terms of shared phenomena:* forse alle volte sono entrato troppo nel dettaglio e ho espresso machine phenomena? Ho ripensato alcuni requirements che mi sembravano solo machine phenomena, li ho segnati con dei commenti, la cosa importante secondo me è essere coerenti (anche perché quello che succede tra server e applicazione è shared o soltanto machine?)
2. Alle volte ho paura di stare entrando troppo nell’implementazione, che invece andrebbe evitata in un RASD.

**OPEN QUESTIONS**

* What happens if the number of people satisfying the request is bigger than 1000 when the request is accepted, but then decreases under this threshold (due to death, aging etc.)?

[qua dobbiamo metterci d’accordo]

*La mia proposta: se il numero di persone che corrisponde a quella query scende sotto i mille, i dati non vengono più resi disponibili alla third party (lo trovo più semplice da modellare con i goals)*

* How to distinguish between third parties in order to provide unique identification?

*La mia proposta: indirizzo e-mail ufficiale della società.*

(per le persone usiamo il fiscal code)

* + Che cosa dobbiamo modellizzare con Alloy? Brain storming: data, third party, user, application, queries, requests…..

NB

* Data for help fa back up notturno, Automated SOS fa back up ogni ora
* L’utente ha la possibilità di disattivare temporanemente Automated SOS manulamente
* Il primo che se ne prende in carica
* Dobbiamo dire che distinguamo i registration form per paziente e third company
* Waterproof ma fino ad un certo punto
* Se per un minuto non leggo i dati bene dal sensore, chiedo se sta bene