Domotic Room Server

Benini - Casadei - Benedetti

University of Bologna

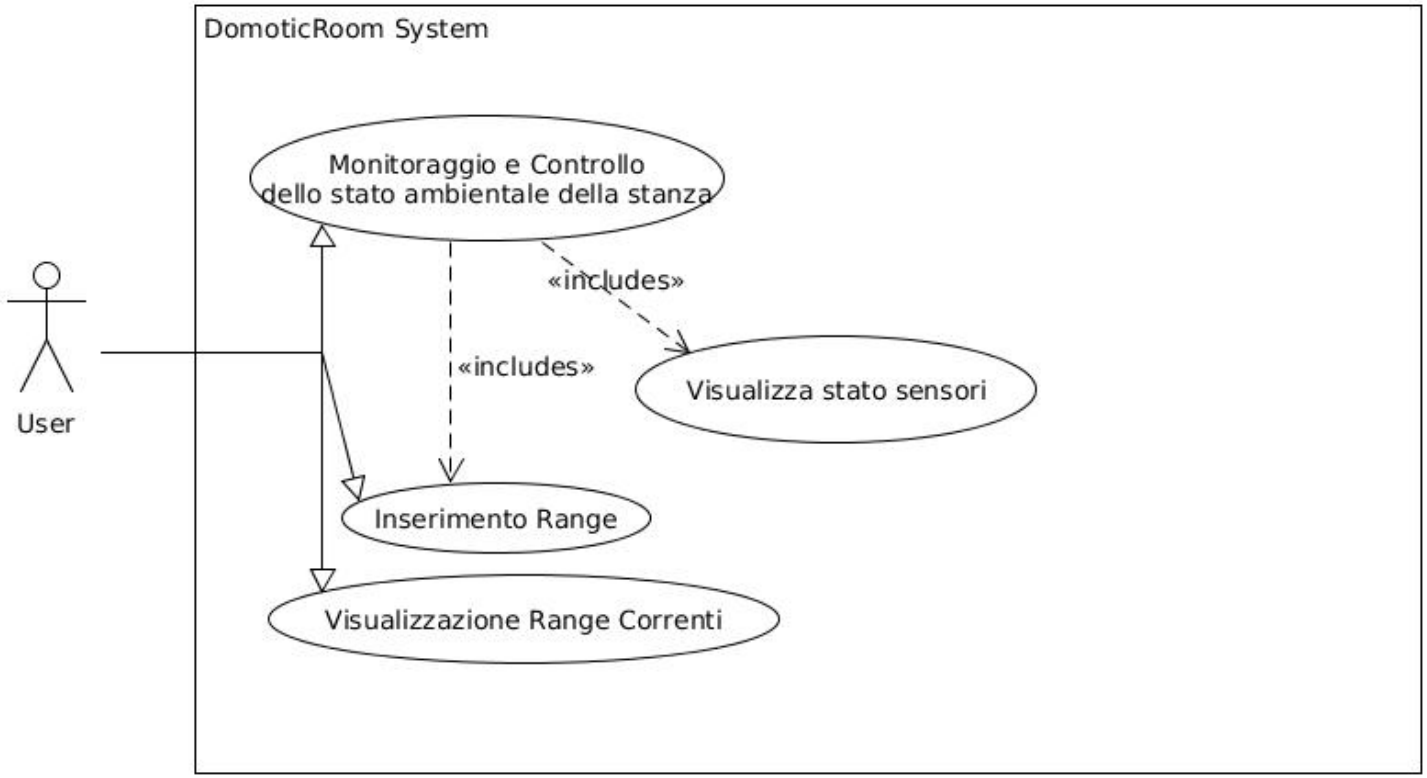
e.benini5@studio.unibo.it

April 24, 2016

Overview

Use Cases

Utilizzo principale della parte server.



Domain Model - Structures

Entitá principali riguardanti il dominio applicativo

- Sensori: coloro che inviano i dati al server.
- Ranges: Intervalli su cui controllare la validitá dei dati raccolti.

Entitá dai requsiti non funzionali

- PersistenceStore: Dove salvare i dati.
- Analizer: computa analisi sui dati.

Domain Model - Interaction and Behaviour

Prime considerazioni riguardo l'interazione

- *Polling:* come metodo per ottenere dati dai sensori.
- *Procedure Call:* Viene invocata dall'estrerno una funzionalitá del sistema.

Prime considerazioni riguardo il comportamento

- *Wait:* Attesa dell'arrivo di una richiesta.
- *Compute:* Viene effettuata una computazione sulla base della richiesta
- *Output:* Viene mostrato il risultato.

Abstraction Gap

Paradigma

L'utilizzo di un'interazione a polling é molto primitivo ed é preferibile, per questioni di design del codice e mantenibilitá, passare ad un paradigma che meglio modella il nostro caso di studio, il **paradigma reactive**

Reattivitá

Le motivazioni che hanno portato alla scelta del paradigma reactive risiedono nelle proprietá che il [Reactive Manifesto] presenta, che coincidono con i requisiti che vogliamo garantire nel sistema.

Interoperabilitá

É necessario che il server sia indipendente dal client a meno di un protocollo interoperabile tra loro, di conseguenza si é scelto di effettuare le comunicazioni attraverso il **protocollo HTTP e il formato JSON.**

Logic Architecture - Structure

Packages:

- *Persistence:* Contiene tutto ció che riguarda li salvataggio dati.
- *StreamBuilder:* Contiene entitá che costruiscono gli steam che trattano i dati in input/output
- *Manager:* effettuano computazioni sui dati, come ad esempio il controllo della coerenza con i range attivi.
- *DataFormatter:* trasformano i dati in ingresso in rappresentazioni interne e viceverse.

Alcuni dei pacchetti presenti nell'architettura logica sono stati omessi visto che, in fase di progetto sono stati sostituiti principalmente per motivi tecnologici e per semplificare l'architettura. Ció non significa che siano inutili perché hanno contribuito al processo di analisi e possono essere utili come base per altri progetti simili.

Logic Architecture - Interaction and Behaviour

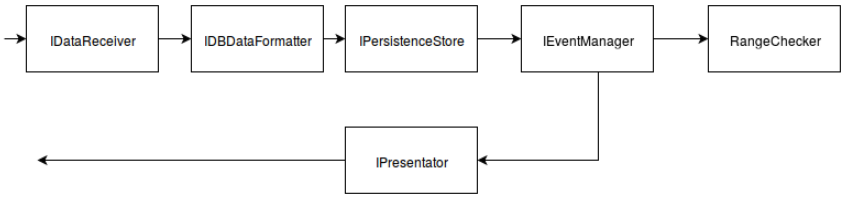


Figure: New Range Stream

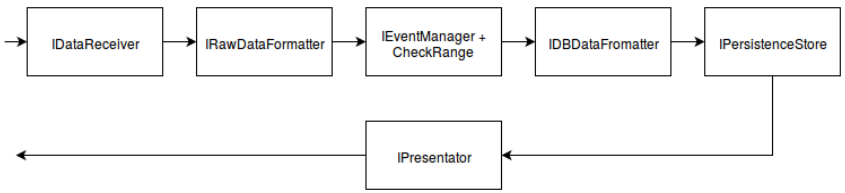


Figure: Incoming Data Stream

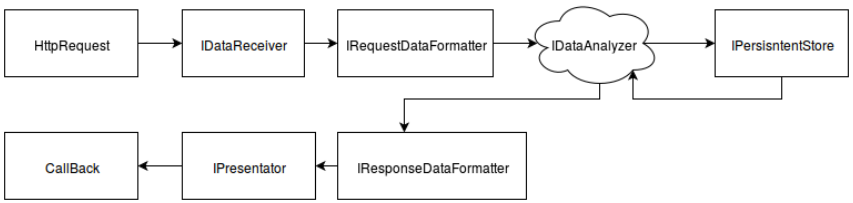


Figure: Request Data Stream

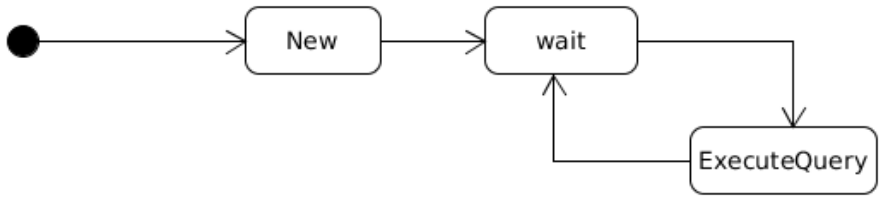


Figure: Persistence Store Behaviour

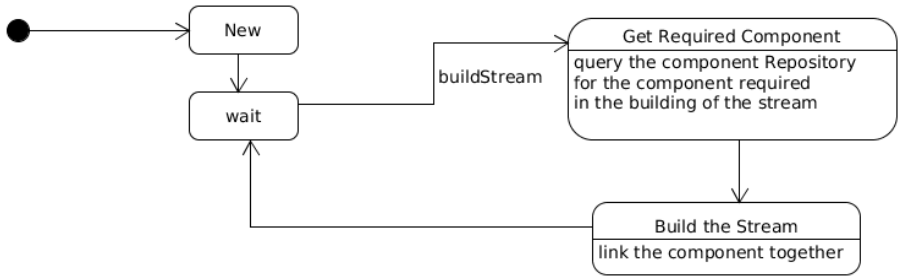


Figure: Stream Builder Behaviour

Technologies and Tools

Bootstrap

Project Architecture Differences

- *Web Pages:* vengono utilizzate come output dell'elaborazione, grazie anche al supporto del play framework.
- *Startup:* era stata prevista una classe che si occupasse della creazione dei vari elementi del sistema, mentre invece questo viene effettuato dal framework.
- *Data Analisys:* per computazioni di analisi sui dati erano stati previsti degli oggetti specifici, mentre, per semplicitá si é demandato tutto a primitive del DBMS (average, max, min).
- *Rappresentazione Interna delle Entitá di Dominio:* Per tutte le entitá di dominio interne al sistema é stata pensata una rappresentazione interna per rendere indipendende il sistema stesso dalle rappresentazioni provenienti dall'esterno e quelle necessarie per il persistence store. Sono state necessarie anche l'implementazione di operazioni di conversione.

References

Reactive Manifesto

http://www.reactivemanifesto.org/

Reactive Extensions

http://reactivex.io/

Play Framework

https://www.playframework.com/

Scala Language

http://www.scala-lang.org/

MongoDB

https://www.mongodb.org/

Akka Framework

http://akka.io/

Bootstrap

http://getbootstrap.com/