ISSN/0393-3016

F. COLAO, S. DI FRISCHIA

Divisione Tecnologie Fisiche per la Sicurezza e la Salute
Laboratorio Diagnostiche e Metrologia
Centro Ricerche Frascati, Roma

PROGETTAZIONE E REALIZZAZIONE DI UN SISTEMA INFORMATIVO PER L'ACQUISIZIONE E L'ANALISI DEI DATI LIDAR RACCOLTI NELL'AMBITO DEL PROGETTO RITMARE

RT/2017/20/ENEA

AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE,
L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

F. COLAO, S. DI FRISCHIA

Divisione Tecnologie Fisiche per la Sicurezza e la Salute
Laboratorio Diagnostiche e Metrologia
Centro Ricerche Frascati, Roma

PROGETTAZIONE E REALIZZAZIONE DI UN SISTEMA INFORMATIVO PER L'ACQUISIZIONE E L'ANALISI DEI DATI LIDAR RACCOLTI NELL'AMBITO DEL PROGETTO RITMARE

RT/2017/20/ENEA

AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE,
L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

PROGETTAZIONE E REALIZZAZIONE DI UN SISTEMA INFORMATIVO PER L'ACQUISIZIONE E L'ANALISI DEI DATI LIDAR RACCOLTI NELL'AMBITO DEL PROGETTO RITMARE

Francesco Colao, Stefano Di Frischia

Riassunto

Nell'ambito del programma RITMARE, l'ENEA ha sostenuto lo sviluppo di un sistema informativo su web per l'acquisizione, la gestione, l'analisi e il reporting dei dati LIDAR raccolti durante le campagne oceanografiche. In particolare, i principali obiettivi sono stati individuati nei seguenti task: progettare un sistema informatico collegato al sensore LIDAR per salvare e visualizzare i dati in tempo reale, per effettuare analisi approfondite dei dati raccolti, per permettere il download di dati e informazioni in diversi formati. Il sistema inoltre accede ad un repository unico dei dati di missioni diverse per confronti e analisi comparate. Nel report si analizzano i requisiti funzionali e le specifiche supplementari adottate durante la fase di progettazione e sviluppo, che ha portato all'implementazione di un web server con architettura multi-livello (presentazione, modello logico, base di dati) a cui l'utente accede tramite una facile interfaccia web strutturata a diversi livelli di utenza. Nel report sono mostrate le varie fasi di progettazione, il modello dei dati, ed alcuni esempi di report di analisi elaborati dal sistema su dati reali raccolti durante diverse campagne.

Parole chiave: software, server, architettura web multi livello, lidar fluorosensore, campagne oceanografiche.

Abstract

In the frame of the Italian the RITMARE program, ENEA has supported the development of a web information system for the acquisition, management, analysis and reporting of LIDAR data collected during oceanographic campaigns. RITMARE is a Flagship initiative launched by the Italian Ministry of University and Research under the National Research Programs responsibility, and it involves an integrated effort of most of the scientific community working on marine and maritime issues, as well as some major industrial groups. The Diagnostic and Metrology laboratory is involved in WP2 of Subproject 5 (Observational systems based on remote sensing data) to develop a LIDAR system for marine parameters measurements. In details, the main objectives were identified in the following tasks: designing a computer system connected to the LIDAR sensor to save and display the data in real time to perform in-depth analysis of the collected data to allow downloading of data and information in different formats. The system also allows to access a unique repository of previous mission data for data comparisons analyzes. The report analyzes the functional requirements and additional specifications adopted during the design and development phase, which led to the implementation of a multilevel web server (presentation, logical model, database) to which the user can have the access via an easy structured web interface at different user levels. The report shows the various designing phases, the data model, and some sample analysis reports processed by the system on real data collected during several campaigns.

Keywords: *Software, multi level web server, Lidar fluorosensor, oceanographic measurements*

INDICE

1. Introduzione 7

2. Formulazione del problema e opportunità 8

3. Stato dell'arte 9

4. Obiettivi a livello dell'utente 11

5. IPOTESI DI ARCHITETTURA COMPLESSIVA 12
   5.1. PROPOSTA 1. Nodi Locali e Client Remoto 12
   5.2. PROPOSTA 2. Nodo Remoto e Client Remoto 12
   5.3. PROPOSTA 3. Nodi Locali, Nodo Remoto e Client Remoto 12

6. MODALITA' DI COMUNICAZIONE FRA DISPOSITIVO E NODO LOCALE (IPOTESI) 13

7. MODALITA' DI COMUNICAZIONE FRA DISPOSITIVO E NODO LOCALE (DETTAGLIO) 14

8. OPERAZIONI: RICHIESTA DELLA STRUTTURA DEL FILE SYSTEM 15

9. OPERAZIONI: RICHIESTA DI UN FILE SPECIFICO 16

10. SPECIFICHE SUPPLEMENTARI 16
   10.1. USABILITA' 16
   10.2. AFFIDABILITA' 16
   10.3. PRESTAZIONI 17
   10.4. SOSTENIBILITA' 17

11. GLOSSARIO 17
   11.1. Terminologia 17

12. MODELLO DEI CASI D'USO 18
   12.1. DIAGRAMMA DEI CASI D'USO 18
   12.2. TESTO DEI CASI D'USO 18
      12.2.1. CASO D'USO UC1: Visualizza Report Dati Lidar Grezzi 18
      12.2.2. CASO D'USO UC2: Visualizza Report Dati Lidar Pre-elaborati 19
      12.2.3. CASO D'USO UC3: Visualizza dati di navigazione 20
      12.2.4. CASO D'USO UC4: Visualizza scheda tecnica del sensore 20

13. MODELLO DI DOMINIO 21

14. ARCHITETTURA SOFTWARE 22

15. DIAGRAMMA DEI PACKAGE 27

16. DIAGRAMMA DI SEQUENZA 30

17. MODELLO DEI DATI 36

18. ANALISI RAPPORTO VALORI FLUORESCENZA 44

19. BIBLIOGRAFIA 45

1. Introduzione

RITMARE è uno dei Progetti Bandiera del Programma Nazionale della Ricerca finanziato dal Ministero dell'Università e della Ricerca. È coordinato dal CNR e riunisce in uno sforzo integrato la comunità scientifica italiana coinvolta in attività di ricerca su temi marini e marittimi, oltre ad una significativa rappresentanza degli operatori privati del settore. Il progetto è iniziato nel 2012 e terminerà nel 2017[1].

Il Progetto RITMARE si propone la finalità di declinare le priorità del cluster marittimo italiano, identificate in coerenza con le vision strategiche ed i programmi europei, nelle seguenti aree tematiche:

- **Tecnologie marittime -** Attività di ricerca sui temi della sicurezza e della security, della sostenibilità ambientale, del comfort, dell'efficienza e dello sviluppo di nuovi materiali, processi e componenti, volta a supportare nel medio periodo il potenziale competitivo dell'industria nazionale, affermando maggiormente l'eccellenza dei prodotti navali e nautici come simbolo del Made in Italy.
- **Tecnologie della pesca sostenibile** – Ricerca e sviluppo di tecnologie avanzate per la gestione delle risorse ittiche in un'ottica di ecosostenibilità della pesca e per la sicurezza in mare.
- **Tecnologie per la gestione sostenibile della fascia costiera** – Ricerca e sviluppo di tecnologie finalizzate ad incrementare la comprensione dei processi di cambiamento e il controllo in tempo reale dei rischi ambientali riguardanti la fascia costiera per l'attivazione di possibili interventi di emergenza, nonchè delle potenziali forme di intervento antropico su di essi a minimo impatto ambientale, nella prospettiva di recupero delle attuali situazioni critiche e di prevenzione di nuove criticità.
- **La rete internazionale dei laboratori per il Mar Mediterraneo** – Istituzione del laboratorio internazionale multinodale dedicato allo studio del Mar Mediterraneo, per la gestione di una rete osservativa di dati ambientali, per la realizzazione di interventi formativi e per l'attuazione di politiche di partenariato scientifico nell'area mediterranea aperto anche alla partecipazione attiva dei Paesi non europei. Aggiornamento dei laboratori nazionali già esistenti sulla base degli standard internazionali e intercollegamento degli stessi per l'avvio di azioni sinergiche e la facilitazione dello scambio di dati
- **L'adeguamento delle infrastrutture nazionali di ricerca** – Sviluppo di interventi di aggiornamento e razionalizzazione della flotta nazionale di navi da ricerca, attraverso refitting e nuove realizzazione di mezzi, in relazione alle necessità operative connesse alla assunzione da parte dell'Italia di un ruolo di prima linea nell'attuazione delle politiche europee in campo marittimo suggerite dal Libro Blu della Commissione Europea e nell'implementazione dei programmi internazionali di ricerca negli oceani e nelle regioni polari. Adeguamento/aggiornamento delle infrastrutture di ricerca già esistenti con particolare riferimento a quelle dedicate alle linee di ricerca convergenti con gli obiettivi di cui al Programma RITMARE.

Gli obiettivi specifici del progetto sono i seguenti:

- supportare politiche integrate per la salvaguardia dell'ambiente (la salute del mare);

- permettere uno sfruttamento sostenibile delle risorse (il mare come sistema di produzione);
- avviare una strategia di prevenzione e mitigazione degli impatti naturali (il mare come fattore di rischio).

Il sottoprogetto SP5 del progetto RITMARE ha come obiettivo lo sviluppo di un LIDAR marino e del sistema di gestione remota dei dispositivi LIDAR, all'interno di una infrastruttura che permetta lo scambio dei dati attraverso differenti canali, la configurazione dei dispositivi, l'interrogazione, la visualizzazione e il reporting dei dati stessi, ed altre operazioni che andremo a definire[2].

Un LIDAR marino utilizza un impulso laser di breve durata per indurre luminescenza a seguito di diffusione anelastica. L'emissione indotta dal laser viene raccolta da un telescopio ottico, filtrata spettralmente ed infine inviata al sensore ottico, che provvede alla conversione del segnale ottico in segnale elettrico. L'informazione sullo stato fisico-chimico del campione in esame è contenuta nell'intensità di radiazione fluorescente distribuita in bande la cui posizione spettrale è specifica per ciascun componente.

2. Formulazione del problema e opportunità

Attualmente, i dati recuperati sul posto da un dispositivo LIDAR sono elaborati localmente e presentati in output all'operatore in un formato analogo a quello di un foglio elettronico (XLS). Data la limitata capacità computazionale e di memoria dell'hardware connesso al sensore LIDAR, non è possibile effettuare *on the fly* un'analisi approfondita dei dati estratti, né di avere un repository o una base di dati con cui eseguire confronti con dati precedenti. Ciò implica una limitata operatività da parte dei ricercatori durante lo svolgersi della missione, la mancanza di un'elaborazione tempestiva delle informazioni raccolte, nonché l'assenza di un'eventuale comunicazione e scambio di dati fra differenti dispositivi LIDAR in missioni diverse (prospettiva futura).

Uno dei problemi centrali resta quello dell'impossibilità di connettersi ai sensori LIDAR da remoto, ossia da parte di operatori non presenti localmente in missione, che, in questo modo, avrebbero la possibilità sia di avere un'efficiente e rapida valutazione delle informazioni raccolte sul campo, sia di poter configurare e modificare il comportamento dei dispositivi attraverso un'interfaccia web.

La soluzione è quella di progettare e realizzare un'infrastruttura di rete che comprenda sensori LIDAR, nodi locali, nodi remoti e client remoti in modo da poter connettere questi elementi in un unico ambiente, che possa facilitare lo scambio, l'analisi, l'interrogazione e il download di dati e informazioni. Inoltre, con la possibilità di poter configurare i sensori da parte di client remoti, si può offrire una maggiore interazione fra operatori presenti sul campo e non, migliorando la qualità e l'efficacia della ricerca.

Un ulteriore obiettivo è quello di definire un formato comune per i dati che saranno scambiati fra un elemento e l'altro, tenendo presente che il paradigma di comunicazione sarà sempre quello *client-server*. Il formato dovrà anche tenere conto della possibilità di immagazzinare i dati in una base di dati, che sia

relazionale o di altro tipo, facendo in modo che il dato sia, allo stesso tempo, elaborabile da un calcolatore e leggibile da un utente medio.

Oltre ai dati raccolti dai dispositivi LIDAR, ci saranno anche i dati di configurazione dei sensori stessi: un elenco di parametri che ne definisce le caratteristiche tecniche e operative. Entrando più nel dettaglio, possiamo dividere i parametri in due gruppi:

- parametri di configurazione variabili che possono essere modificati dagli operatori in caso di necessità
- parametri costanti che non potranno essere modificati.

Questi ultimi in particolare, saranno utilizzati per la configurazione che specifica la modalità operativa per le misure.

3. Stato dell'arte

Al giorno d'oggi, esistono sistemi software che elaborano dati da sensori LIDAR nel campo di applicazione della topologia e della mappatura 3D. In particolare, i due sistemi più interessanti dal nostro punto di vista, sono:

- il LEDAS (Lidar Experimentation and Data Analysis System) che consiste in un sistema estendibile *web-based* progettato per il controllo della strumentazione lidar, l'analisi dei dati e la loro visualizzazione. I componenti del sistema sono un'interfaccia di desktop remota, un sottosistema di gestione del database e un sottosistema di elaborazione dei dati (Figura 1)[3].

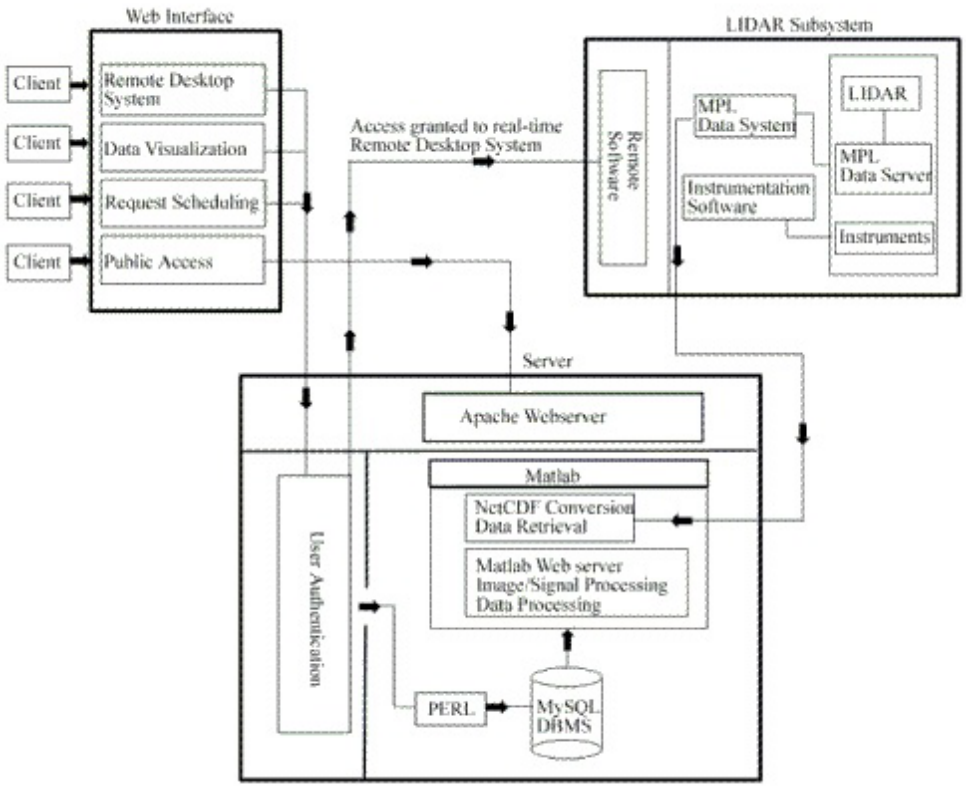


Figura 1 –Architettura LEDAS.

- il sistema implementato per il portale GEON che utilizza un'infrastruttura GRID (a griglia, cioè sfrutta il calcolo parallelo su numerosi calcolatori per aumentare l'efficienza degli algoritmi) per elaborare i dati dei sensori LIDAR. In questo progetto, possiamo tralasciare la struttura a griglia (dato che il nostro sistema non ha bisogno di enormi risorse di calcolo), ma potremmo riprendere invece la struttura a 3 livelli (interfaccia, controllo, elaborazione) proposta (Figura 2)[4].

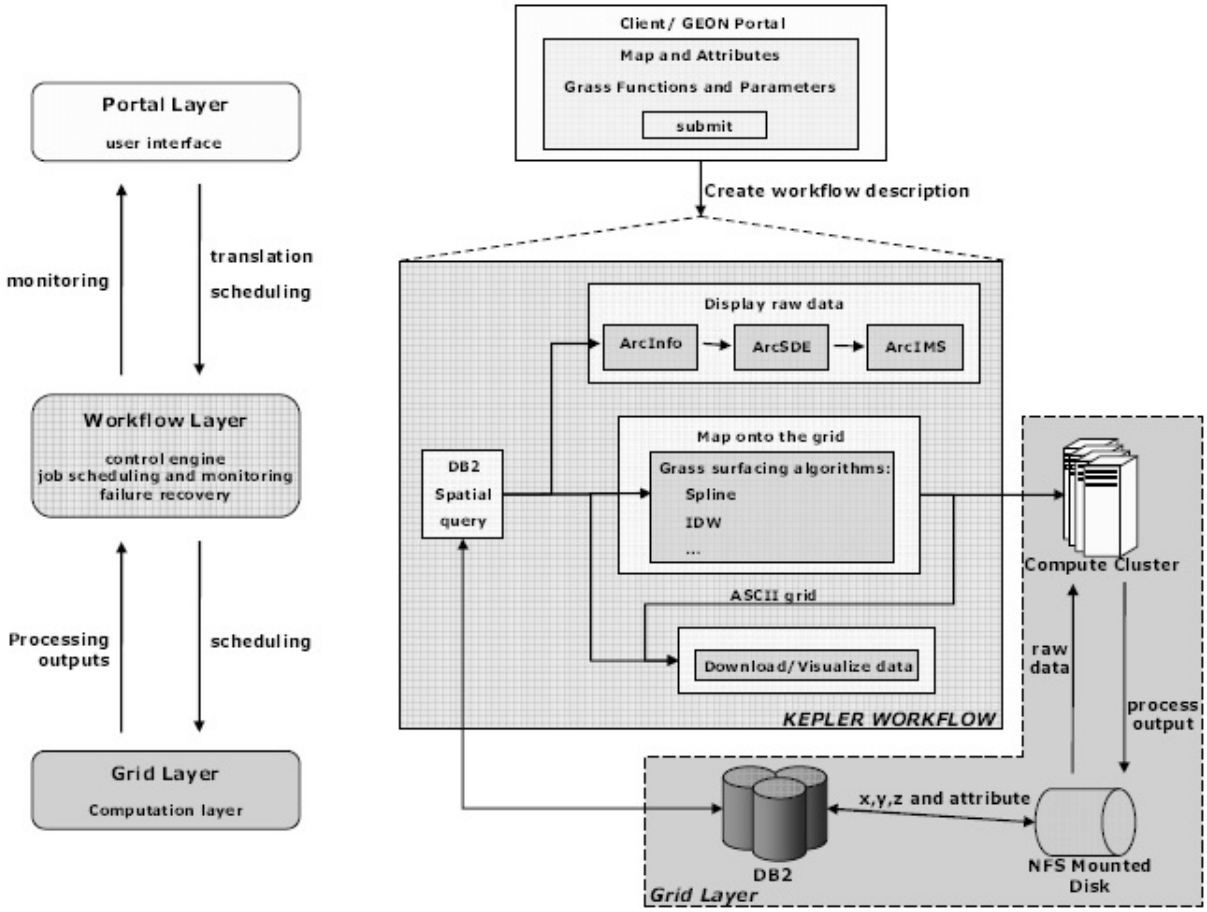


Figura 2 – Architettura di processo lidar a tre livelli

4. Obiettivi a livello dell'utente

Gli utenti (ed eventuali sistemi esterni) necessitano di un sistema a differenti livelli di autenticazione che, in una logica gerarchica, esponga solo le operazioni permesse:

- UTENTE SEMPLICE: visualizzare da locale/remoto un plot o un grafico dei dati raccolti, fare un download dei dati in formati diversi
- UTENTE MANAGER: visualizzare e scaricare dati (vedi Utente Semplice), elaborare i dati grezzi del sensore
- AMMINISTRATORE: stesse operazioni di Utente Manager, e in più gestire account e basi di dati, e in generale tutte le operazioni di natura tecnica del sistema

Ci sarà inoltre una differenza fra le operazioni svolte da Utenti/Operatori IN MISSIONE, e Utenti/Operatori IN REMOTO

5. IPOTESI DI ARCHITETTURA COMPLESSIVA

5.1. PROPOSTA 1. Nodi Locali e Client Remoto

Ogni nodo locale collegato ad un LIDAR andrà a costituire una infrastruttura di rete in cui possiamo ipotizzare almeno due condizioni fondamentali: ogni nodo locale può comunicare con altri nodi locali, e soprattutto ogni nodo locale è accessibile da un client remoto. Questa prima proposta di architettura è mostrata nella Figura 3.

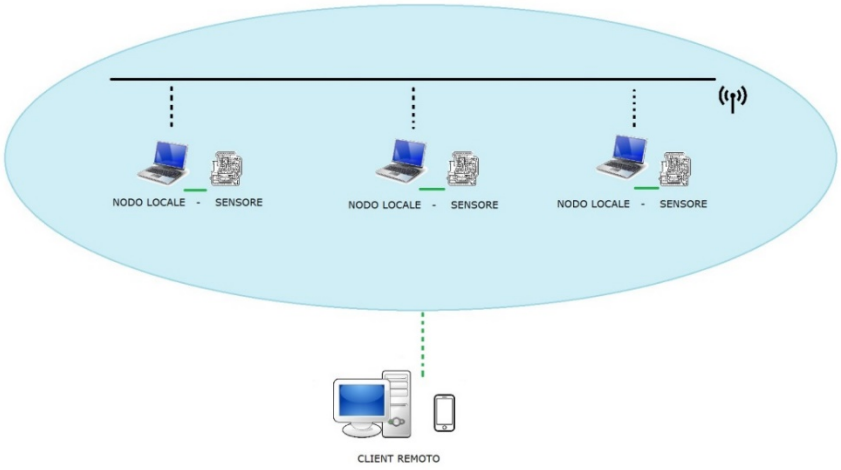


Figura 3 – Proposta 1

5.2. PROPOSTA 2. Nodo Remoto e Client Remoto

I sensori LIDAR invieranno i loro dati direttamente in remoto ad un server (concentratore) che si occuperà dell'elaborazione, e dello scambio dati fra il client remoto e i sensori stessi. L'architettura del sistema cambierà come illustrato nella Figura 4.

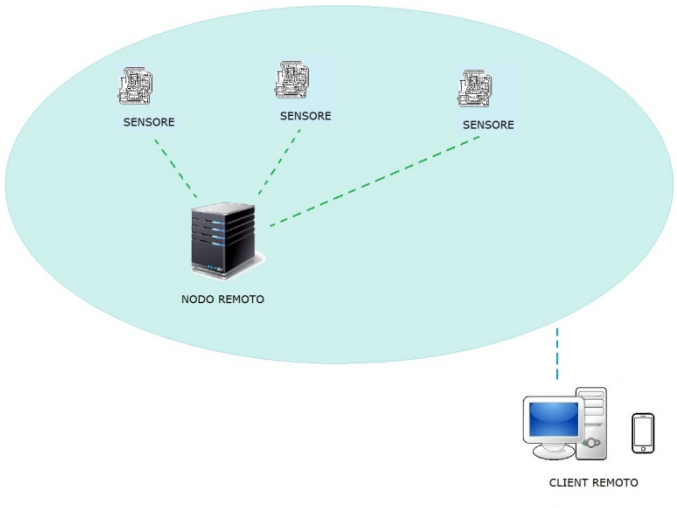


Figura 4 – Proposta 2

5.3. PROPOSTA 3. Nodi Locali, Nodo Remoto e Client Remoto

La soluzione attualmente più logica sembrerebbe quella di una sintesi fra le due proposte precedenti. Ovvero, mantenendo un nodo locale per ogni dispositivo LIDAR affinchè processi i dati grezzi in una prima

elaborazione, e avendo un web server remoto che gestisca la base di dati, che faccia elaborazioni più complesse, gestisca l'autenticazione utenti, eccetera (Figura 5).

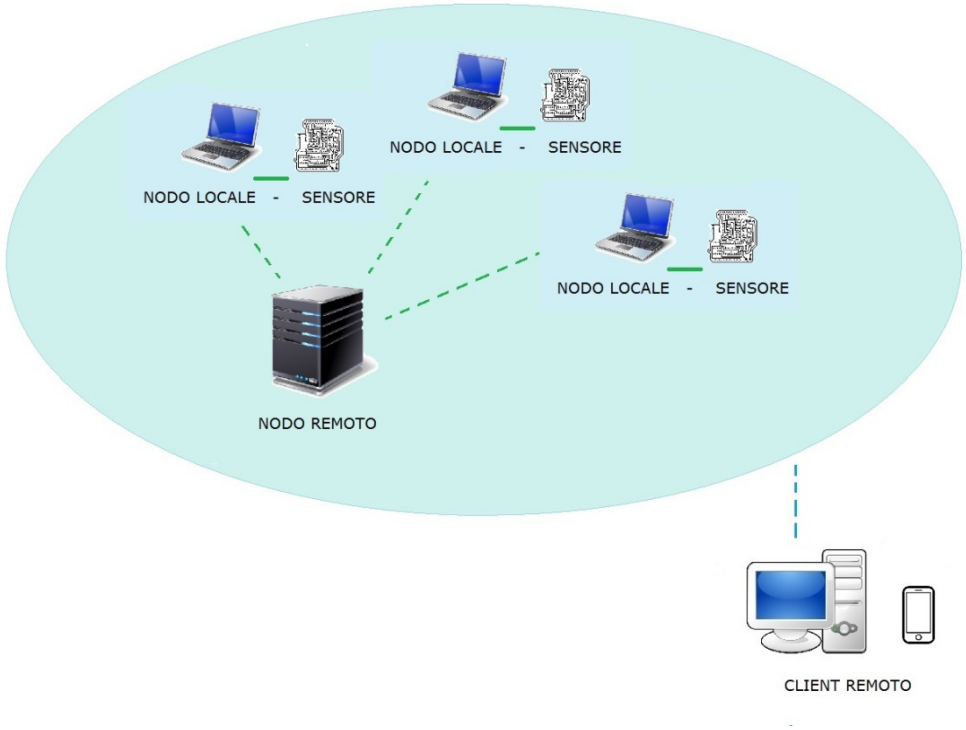


Figura 5 – Proposta 3

6. MODALITA' DI COMUNICAZIONE FRA DISPOSITIVO E NODO LOCALE (IPOTESI)

I dati acquisiti dal sensore LIDAR, che andranno poi trasmessi verso gli altri componenti, sono salvati secondo una struttura analoga a quella di una comune base di dati o di un foglio elettronico. Si dovrà stabilire se gestire l'interrogazione dei dati in modalità passiva o attiva (dal punto di vista del sensore).

In modalità passiva, un nodo locale o remoto collegato con il dispositivo eseguirà un polling e un download dei dati accedendo al dispositivo, mentre in modalità attiva, sarà il sensore ad inviare i propri dati accedendo al nodo più vicino. La decisione fra le due modalità sarà principalmente influenzata dal carico di dati prelevati dal LIDAR (Figura 6).

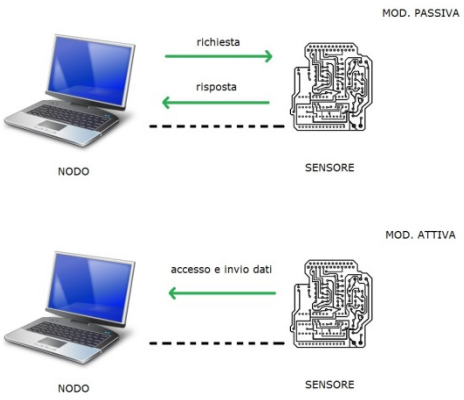


Figura 6 - Modalità di comunicazione fra dispositivo e nodo locale, a sinistra (passivo) mentre a destra (attivo)

7. MODALITA' DI COMUNICAZIONE FRA DISPOSITIVO E NODO LOCALE (DETTAGLIO)

Il problema di far comunicare due processi diversi che risiedono oltretutto su macchine dall'architettura hardware e software profondamente differente, dev'essere gestito con estrema attenzione, analizzando il flusso di dati che dovrà essere elaborato dai due calcolatori, e predisponendo con precisione la gestione degli errori e delle eccezioni.

Da una parte avremo un sistema informativo su web residente su un nodo locale che gestirà i dati 'grezzi' mandati in output dal dispositivo. Questo sistema dovrà quindi eseguire delle operazioni per avere accesso ai dati del dispositivo. In alcuni casi le operazioni potranno essere avviate dagli stessi operatori tramite l'interfaccia web (modifica di alcune impostazioni del LIDAR, download delle misure in un intervallo di tempo specifico, ecc.), in altri casi si rende necessario predisporre delle operazioni da eseguire in automatico e ad intervalli di tempo predefiniti, in maniera totalmente trasparente all'utente finale (ad esempio, il download e il backup dei dati dell'ultima ora).

Dall'altra parte avremo un dispositivo su cui gira un firmware dedicato, in grado di compiere una serie di operazioni, di interfacciarsi con altre componenti hardware e di memorizzare su scheda le misure raccolte dal LIDAR.

Facendo riferimento al precedente paragrafo, la modalità che si è dimostrata più adatta per la serie di operazioni da implementare, è stata quella passiva (dal punto di vista del dispositivo LIDAR). Ovvero il nodo locale, tramite un protocollo di comunicazione stabilito a priori, richiede le informazioni desiderate al dispositivo, che provvederà ad soddisfare la richiesta o, in caso contrario, a lanciare un eccezione da inoltrare al nodo locale.

La traduzione della richiesta generata dal server web verso il dispositivo LIDAR, sarà processata da un componente software intermedio (che chiameremo per semplicità **VEGA**), il cui eseguibile principale si occuperà dell'inoltro e della ricezione di dati e informazioni tra le due parti interessate. In particolare, l'eseguibile avrà come parametro uno script generato dal web server contenente dei comandi predefiniti che saranno tradotti in una modalità accettata dal dispositivo.

Riassumendo, la sequenza tipica di una comunicazione fra sistema web e dispositivo LIDAR (Figura 7) sarà la seguente:

- Il sistema web attiva un sottoprocesso *batch* per l'esecuzione di un'operazione pianificata
- Il sistema web genera un file di script che contiene i comandi nel protocollo di comunicazione predefinito tra i componenti dell'architettura
- Il sistema web lancia l'eseguibile del modulo VEGA, dandogli come parametro lo script appena creato
- Il modulo VEGA processa lo script ed inoltra le richieste al dispositivo, ricevendo come risposta il dato richiesto (tipicamente un file)
- Il sistema web accede al file aggiornato nella sua memoria locale, e lo elabora secondo le esigenze

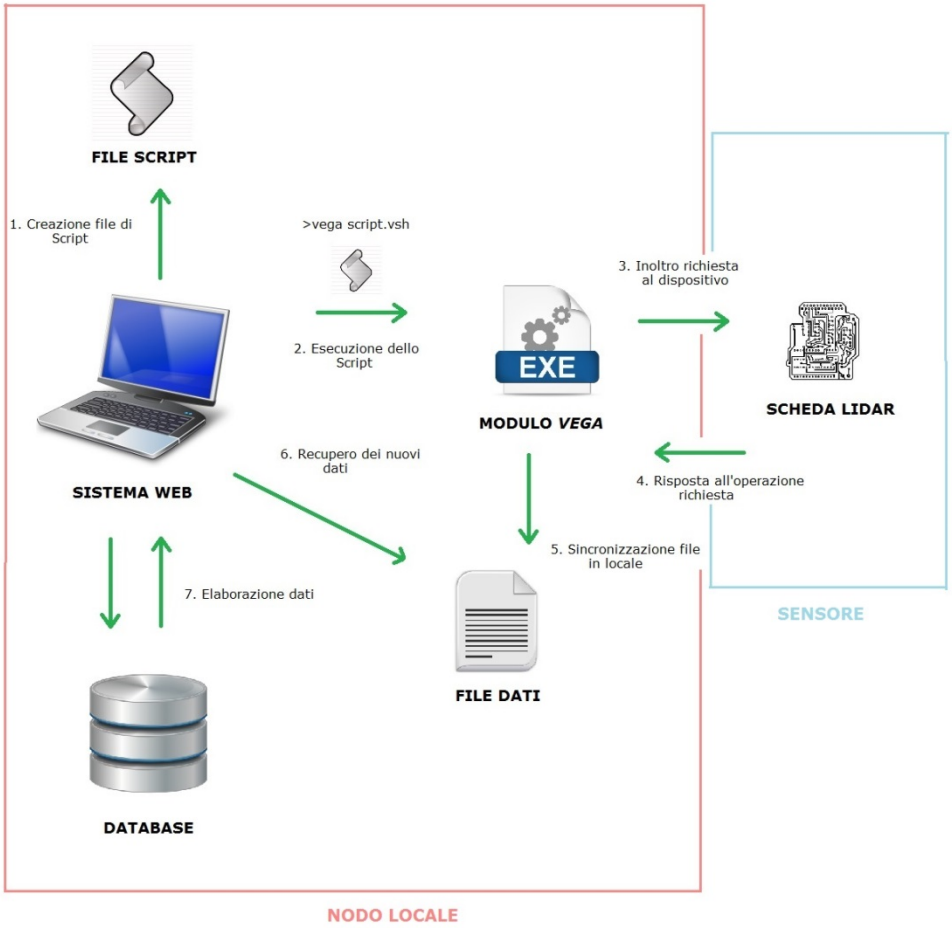


Figura 7 – Modalità di comunicazione fra dispositivo e nodo locale

8. OPERAZIONI: RICHIESTA DELLA STRUTTURA DEL FILE SYSTEM

Una delle operazioni che il sistema web può richiedere al dispositivo, è quella di recuperare un file che rappresenti la struttura del *filesystem* memorizzato nella scheda del dispositivo stesso. Nel nodo locale infatti, ci sarà un *mirror* (ovvero una copia esatta) dei file e delle cartelle presenti nella scheda. In questo modo, il sistema sincronizzerà periodicamente i file dati e di configurazione salvati in locale con quelli del dispositivo LIDAR, avendoli così a disposizione e minimizzando il tempo di accesso alle risorse.

Supponiamo che il sistema web crei uno script per richiedere l'immagine del *filesystem*. All'interno dello script sarà presente un comando **LS** che il modulo VEGA interpreterà e inoltrerà al dispositivo. La risposta verrà scritta in un file in formato JSON, e salvata successivamente sul nodo locale così da essere a disposizione del sistema web. Un esempio del risultato di un'operazione di questo tipo potrebbe essere il seguente:

{"d":"1979-00-30_12:00:00","e":0,"a":16,"p":0,"n":"RITMARE","z":0}

{"d":"2015-22-23_02:22:30","e":1,"a":16,"p":0,"n":"STR1","z":0}

{"d":"2015-37-24_10:37:28","e":2,"a":32,"p":1,"n":"config.jsn","z":538}

{"d":"2015-22-23_02:22:30","e":3,"a":16,"p":1,"n":"DATA","z":0}

{"d":"2015-22-23_02:22:30","e":4,"a":16,"p":3,"n":"SECT01","z":0}

{"d":"2015-41-05_09:41:00","e":5,"a":32,"p":4,"n":"STR10000.dat","z":6037454}

Le coppie nome-valore rappresentano le seguenti informazioni:

- **d** è la data di ultima modifica del file/directory
- **e** è un identificativo del file/directory
- **a** serve ad indicare se si tratta di un file o di una directory
- **p** indica l'identificativo del nodo genitore
- **n** è il nome del file/directory
- **z** è la dimensione (in byte) su disco

9. OPERAZIONI: RICHIESTA DI UN FILE SPECIFICO

Una volta che il sistema possiede l'immagine del filesystem del dispositivo, è in grado di richiedere un file specifico per le proprie elaborazioni. All'interno dello script da eseguire, sarà quindi presente un comando **GET -idFile**, dove l'identificativo del file, non è altro che il valore di "**e**" del file immagine del paragrafo precedente.

Se l'operazione sarà andata a buon fine, il nodo locale avrà, nel mirror del filesystem, il file richiesto correttamente aggiornato e nella posizione esatta in cui si trova sul dispositivo.

10. SPECIFICHE SUPPLEMENTARI

10.1. USABILITA'

Il sistema deve presentare un'interfaccia efficace ed allo stesso tempo user-friendly. La gestione del dispositivo LIDAR, sia a livello di configurazione, sia a livello di visualizzazione e recupero dati, deve poter avvenire attraverso un'interfaccia grafica comprensibile anche da un utente medio, senza bisogno di scrivere codici o linee di comando. La visualizzazione delle immagini e dei dati, e il loro eventuale download, dev'essere implementato in modo da poter esportare queste informazioni in formati comprensibili a tutte le tipologie di utenti (immagini come file JPEG o PNG, dati come file di testo o fogli Excel, ad esempio). E' auspicabile implementare funzionalità di help all'interno del sistema, per assistere l'utente durante le operazioni.

10.2. AFFIDABILITA'

Il sistema deve poter contare innanzitutto su un insieme integro e corretto di dati. La maggior parte delle operazioni di controllo su di essi, verranno eseguite dalla scheda collegata al dispositivo LIDAR che, al suo interno, esegue una prima analisi dei dati grezzi che saranno inviati al nodo locale.

Un controllo severo sarà anche eseguito nel momento di una riconfigurazione del sensore da parte di un operatore. Nell'eventualità in cui verranno inseriti dei parametri non conformi alle specifiche, il sistema bloccherà la configurazione segnalando all'utente l'errore.

Per quanto riguarda l'affidabilità dell'infrastruttura di rete che collegherà gli elementi sul campo e quelli in remoto, dovranno essere previsti meccanismi di fault tolerance e ritrasmissione.

10.3. PRESTAZIONI

La mole di dati iniziale con cui andranno elaborate le prime operazioni del sistema, non sembra tale da poter generare problemi sull'uso di risorse hardware o di calcolo. I tempi di risposta di connessioni e sessioni da remoto, dovranno essere verificate al momento dell'operatività a regime del sistema.

10.4. SOSTENIBILITA'

Il sistema dovrà essere progettato e realizzato in maniera modulare ed estendibile, in modo da poter successivamente aggiungere operazioni, tipi di dati e funzionalità in genere, rispetto ai casi d'uso previsti inizialmente. Un importante punto da tenere presente per il futuro, è la possibile aggiunta di un repository remoto dei dati storici delle missioni, così come della presenza contemporanea di più sensori che operano contemporaneamente in missioni diverse.

11. GLOSSARIO

11.1. Terminologia

- Il **dispositivo LIDAR** è l'elemento che acquisisce fisicamente i dati sul campo. Ipotizziamo uno scenario in cui esistono diversi sensori LIDAR che compiono operazioni differenti, indipendentemente l'uno dall'altro. In questo caso, ogni sensore dovrà poter essere configurato autonomamente in base al tipo di missione che andrà a svolgere.
- Per **nodo locale**, intendiamo un elemento del sistema presente in prossimità del sensore LIDAR, con cui gli operatori sul campo possono gestire il sensore, ed eventualmente compiere altre operazioni. Assumiamo che quest'elemento sia costituito da un PC fisso o portatile collegato con un qualche tipo di canale (tipicamente un cavo Ethernet) al sensore.
- Per **nodo remoto**, che chiameremo anche **concentratore**, intendiamo invece una macchina che si interfaccia direttamente con i sensori LIDAR da remoto, scaricando e salvando i dati raccolti da ogni dispositivo. Il concentratore avrà le caratteristiche di un web server (o meglio, di un **web service**) più che di un normale PC.
- Un **web service** è un sistema software progettato per supportare l'interoperabilità tra diversi elaboratori su di una medesima rete, ovvero in un contesto distribuito. Lo scopo dei web services è quello di offrire un servizio "all'esterno" verso altri sistemi (anche molto diversi fra loro) tramite un'interfaccia standard, che nasconde l'implementazione interna del web service.
- Per **client remoto**, intendiamo un PC, uno smartphone, o un qualunque altro dispositivo che permetta di accedere e di scaricare i dati prelevati dai sensori LIDAR.

12. MODELLO DEI CASI D'USO

12.1. DIAGRAMMA DEI CASI D'USO

Nel seguente diagramma vengono mostrate le operazioni principali che gli utenti del sistema hanno la possibilità di compiere all'interno di esso. Ogni operazione elencata nel diagramma va considerata come macro-operazione: ognuna di loro può essere infatti scomposta in più operazioni elementari che verranno analizzate nel dettaglio nei diagrammi di sequenza.

Come già spiegato in precedenza, le utenze del sistema saranno suddivise in tre ruoli organizzati in modo gerarchico, e precisamente, partendo dal più basso: END-USER, USER-MANAGER, ADMIN. Nel diagramma di Figura 8, per motivi di chiarezza sono mostrati solo i primi due. L'utente ADMIN possiede le credenziali per compiere tutte le operazioni concesse agli altri due ruoli e, in più, gestisce gli account del sistema, la base di dati, e varie operazioni di natura tecnica e gestionale.

Figura 8 – Diagramma dei ruoli organizzativi.

12.2. TESTO DEI CASI D'USO

Nel seguito analizzeremo alcuni dei casi d'uso principali, che durante la fase di analisi, sono stati ritenuti determinanti per il funzionamento del sistema e il soddisfacimento dei requisiti iniziali

12.2.1. CASO D'USO UC1: Visualizza Report Dati Lidar Grezzi

- Portata: Sistema RITMARE
- Livello: Obiettivo Utente
- Attore Primario: Operatore (con ruolo END_USER o superiore)
- Parti Interessate ed Interessi:

    - Operatore: vuole visualizzare e/o scaricare i dati grezzi secondo certi parametri
    - Server: elabora i dati in memoria per renderli in maniera grafica all'operatore
- Pre-condizioni: L'operatore è identificato ed autenticato nel sistema
- Post-condizioni: L'operatore visualizza una serie di immagini delle misure desiderate e/o ha scaricato un file dati delle misure desiderate

Scenario principale di successo:

1. L'operatore si è connesso al sistema per la visualizzazione
2. L'operatore sceglie la missione desiderata e il sensore relativo
3. Il sistema mostra all'operatore un'interfaccia grafica con la scelta dei canali da consultare, dei parametri temporali e del tipo di grafico
4. L'operatore seleziona i canali desiderati, la finestra temporale e:
    a. La modalità "Grafico" in cui i valori vengono mostrati in diagrammi x-y (tempo-valore)
    b. La modalità "Mappa" in cui i valori vengono mostrati su una mappa secondo una scala di colori, con le coordinate relative
5. Il sistema apre una nuova pagina web con i risultati della ricerca
6. L'operatore può salvare sul proprio supporto le immagini generate dal sistema
7. L'operatore può salvare sul proprio supporto un file di testo contenente tutti i dati dell'intervallo di tempo selezionato

12.2.2. CASO D'USO UC2: Visualizza Report Dati Lidar Pre-elaborati

- Portata: Sistema RITMARE
- Livello: Obiettivo Utente
- Attore Primario: Operatore (con ruolo END_USER o superiore)
- Parti Interessate ed Interessi:
    - Operatore: vuole visualizzare e/o scaricare i dati pre-elaborati secondo certi parametri
    - Server: elabora i dati in memoria per renderli in maniera grafica all'operatore
- Pre-condizioni: L'operatore è identificato ed autenticato nel sistema
- Post-condizioni: L'operatore visualizza una serie di immagini delle misure desiderate

Scenario principale di successo:

1. L'operatore si è connesso al sistema per la visualizzazione
2. L'operatore sceglie la missione desiderata e il sensore relativo
3. Il sistema mostra all'operatore un'interfaccia grafica con i seguenti parametri: la finestra temporale, il tempo di media (in secondi) dell'analisi, l'elenco dei rapporti fra canali (tipicamente fra un canale e il canale 404 di riferimento) e il tipo di grafico
4. L'operatore seleziona i rapporti che vuole visualizzare e gli altri parametri di analisi
5. Il sistema apre una nuova pagina web con i risultati della ricerca

6. L’operatore può salvare sul proprio supporto le immagini generate dal sistema

12.2.3. CASO D’USO UC3: Visualizza dati di navigazione

- Portata: Sistema RITMARE
- Livello: Obiettivo Utente
- Attore Primario: Operatore (con ruolo END_USER o superiore)
- Parti Interessate ed Interessi:
    - Operatore: vuole visualizzare i dati di navigazione su una mappa
    - Server: elabora i dati in memoria per renderli in maniera grafica all’operatore
- Pre-condizioni: L’operatore è identificato ed autenticato nel sistema
- Post-condizioni: L’operatore visualizza una serie di immagini dei dati desiderati

Scenario principale di successo:

1. L’operatore si è connesso al sistema per la visualizzazione
2. L’operatore sceglie la missione desiderata e il sensore relativo
3. Il sistema mostra all’operatore un’interfaccia grafica con una scelta fra alcuni parametri di navigazione (velocità, direzione, ecc…) e la finestra temporale
4. L’operatore seleziona i parametri desiderati
5. Il sistema apre una nuova pagina web con i risultati della ricerca utilizzando i seguenti servizi:
    a. Se il sistema è connesso ad Internet, verrà visualizzata una mappa da Google Maps
    b. Altrimenti, verrà disegnato un diagramma x-y in cui sono rappresentate le coordinate

12.2.4. CASO D’USO UC4: Visualizza scheda tecnica del sensore

- Portata: Sistema RITMARE
- Livello: Obiettivo Utente
- Attore Primario: Operatore (con ruolo END_USER o superiore)
- Parti Interessate ed Interessi:
    - Operatore: vuole visualizzare i dati tecnici di un sensore
    - Server: elabora i dati in memoria per restituire le informazioni all’operatore
- Pre-condizioni: L’operatore è identificato ed autenticato nel sistema
- Post-condizioni: L’operatore visualizza un elenco dei parametri tecnici desiderati e/o scarica un file contenente l’elenco stesso

Scenario principale di successo:

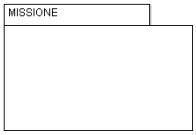
1. L’operatore si è connesso al sistema per la visualizzazione
2. L’operatore sceglie la missione desiderata e il sensore relativo
3. L’operatore accede alla sezione “Scheda Sensore” del sistema
4. Il sistema mostra all’operatore l’elenco dei parametri del sensore, e l’elenco dei canali con i parametri specifici per ognuno di essi

5. L’operatore può salvare sul proprio supporto un documento contenente la scheda tecnica completa del sensore selezionato

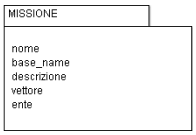
13. MODELLO DI DOMINIO

Il seguente modello di dominio (Figura 9) illustra un diagramma in cui sono presenti cinque elementi grafici:

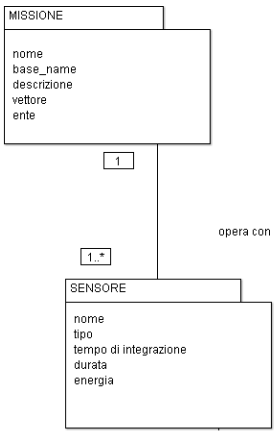
- Classe concettuale: rappresenta oggetti del mondo reale che sono presenti nella logica del sistema



- Attributi delle classi: rappresentano caratteristiche e/o parametri precisi di una classe concettuale



- Associazioni: sono relazioni fra classi (più precisamente tra istanze di queste classi) che rappresentano una connessione significativa e interessante
- Notazione delle associazioni: ogni associazione (indicata con una linea che congiunge due classi) ha
    - un nome, generalmente una locuzione verbale, che permette di comprendere il significato del collegamento. Se non indicato diversamente, si legge l’associazione da sinistra verso destra e dall’alto verso il basso.
    - una molteplicità che definisce quante istanze della classe A possono essere associate alla classe B (uno a uno, uno a molti, eccetera)



ES.: un’istanza della classe Missione ‘opera con’ una o più istanze della classe Sensore. Viceversa un Sensore può essere associato ad una ed una sola Missione

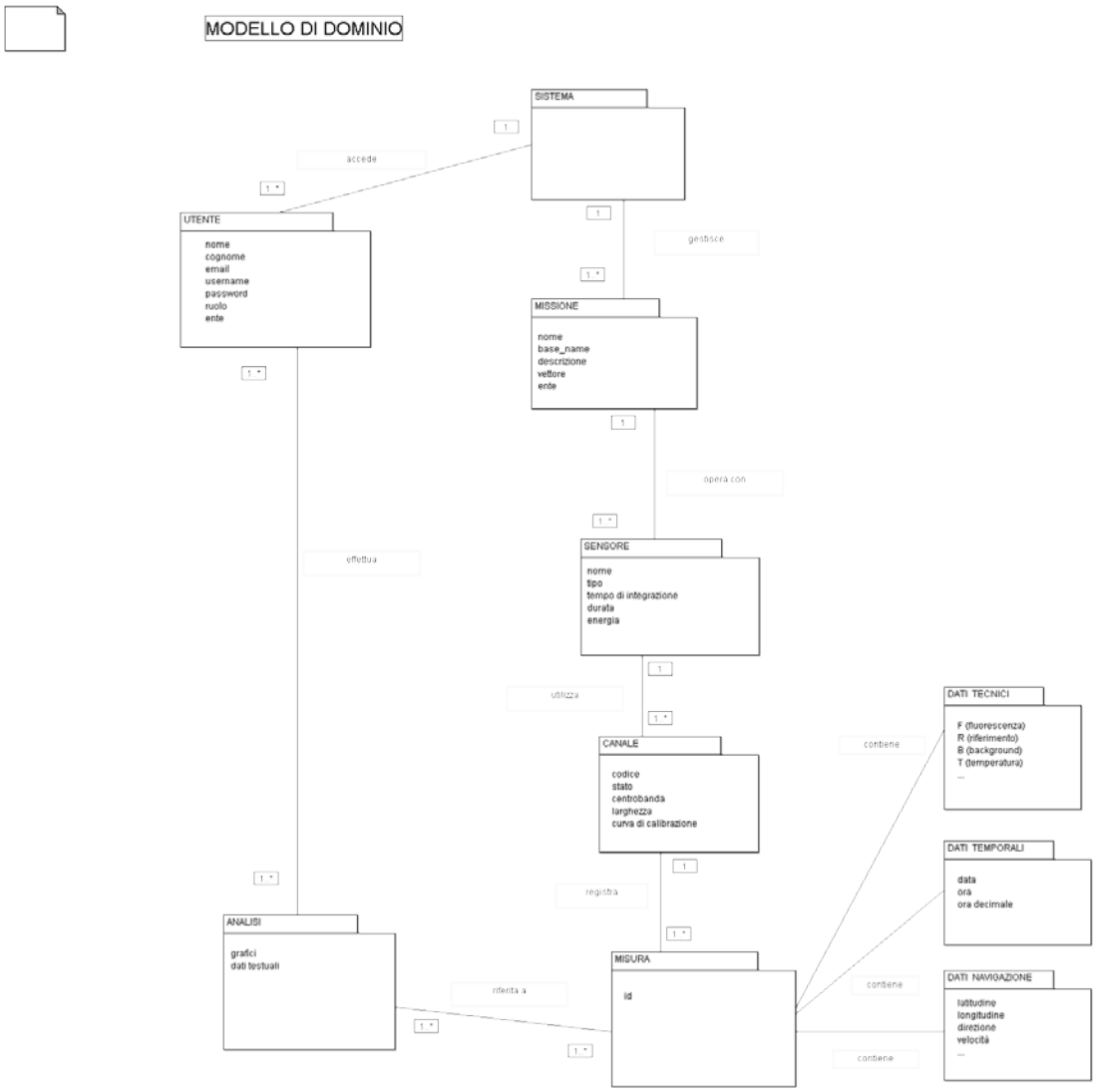


Figura 9 – Diagramma del modello di dominio

14. ARCHITETTURA SOFTWARE

14.1. MODELLO CLIENT-SERVER

Dato che si sta trattando la progettazione di un sistema che offre un servizio ad un insieme di utenti, il primo modello a cui fare riferimento è naturalmente quello client-server. Con questa definizione si intende un'architettura generale del sistema in cui sono presenti uno o più fornitori di un servizio (chiamati server), e i richiedenti del servizio stesso chiamati client. In questo contesto, quasi sempre i client e i server risiedono su hardware separati, comunicando attraverso una rete. Il server ospita una o più applicazioni che condividono le risorse con i clienti. Il cliente, al contrario, non condivide alcuna sua risorsa, ma richiede il servizio desiderato al server per mezzo di precisi protocolli di richiesta-risposta.

14.2. ARCHITETTURA MULTI-TIER

L’architettura multi-tier (multi-strato) è un’architettura client-server in cui le funzioni di presentazione, logica dell’applicazione, e gestione dei dati sono fisicamente separate. Nel nostro caso, parleremo di architettura 3-tier. Nella Figura 10 è rappresentato un esempio tipico di architettura a 3 strati[5].

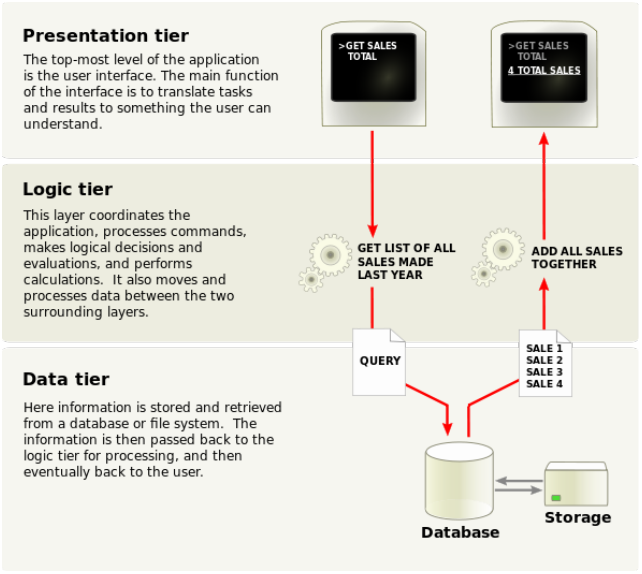


Figura 10 – Esempio di architettura a tre stadi.

Nel sistema informativo RITMARE che andremo a realizzare i tre livelli saranno così suddivisi:

- Strato di Presentazione: interfaccia formata da pagine web in cui l’utente ha a disposizione le operazioni permesse dalle sue credenziali e i risultati delle analisi svolte
- Strato di Logica: moduli software che formano il nucleo del sistema, implementando tutte le operazioni che il sistema offre e coordinando i processi tra i diversi strati, così come la comunicazione tra il sistema e ulteriori device
- Strato dei Dati: formato sia dal File System in cui saranno salvati i file dati grezzi e i file di configurazione dei sensori, sia dal Database in cui questi dati grezzi saranno strutturati in modo da poter essere gestiti dal sistema

14.3. MODEL-VIEW-CONTROLLER

Nella costruzione del nostro sistema informativo su web, il pattern architetturale MVC (Model-View-Controller) sarà quello principale. Per pattern si intende una soluzione architetturale generale e riusabile nei confronti di un problema ricorrente in un dato contesto software.

Nel nostro caso, avendo già chiarito come l’utente avrà a disposizione un’interfaccia web per interagire con il sistema, il pattern MVC (Figura 11) ci viene in aiuto nel separare la rappresentazione interna delle informazioni, dai modi con cui queste informazioni saranno presentate all’utente[6].

Le tre macro-componenti in cui il sistema è idealmente suddiviso sono:

- CONTROLLER: accetta gli input e li converte in comandi per il Modello o la Vista. In particolare, può mandare comandi al Modello per aggiornare lo stato del modello, o alla Vista per cambiare la presentazione del modello nell'interfaccia utente.
- MODELLO: cattura il comportamento del sistema in termini di logica di dominio, indipendentemente dall'interfaccia utente. Il Modello gestisce direttamente i dati, la logica e le regole del sistema. In particolare, il Modello gestisce i dati intercettati dal Controller e mostrati nella Vista
- VISTA: è la rappresentazione in output di qualsiasi informazione gestita dal sistema. In particolare, la Vista riceve informazioni dal Modello attraverso il Controller e le usa per generare un output verso l'utente.

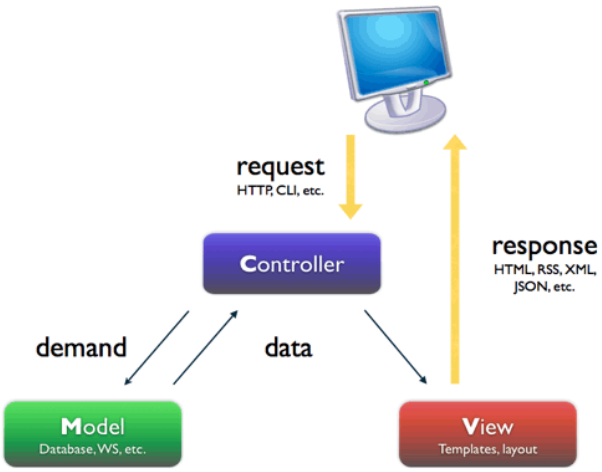


Figura 11 – Schema del pattern MVC.

14.4. FRONT CONTROLLER

E' un pattern architetturale che sviluppa l'idea del Controller presente nel pattern MVC. Più precisamente, un'architettura software che si basa su un Front Controller implica che il sistema fornisca un punto di ingresso centralizzato per la gestione delle richieste.

Nel nostro caso quindi, qualsiasi richiesta proveniente dall'interfaccia utente (tipicamente richieste HTTP generate dai browser dei vari client) verrà catturata dal Front Controller che, consultando una mappa di coppie nome-valore, reindirizzerà la richiesta al modulo software relativo, tipicamente attivando dei Controller specifici che comunicano direttamente con i componenti del Modello del sistema. Nella Figura 12 è schematizzato il comportamento di un Front Controller.

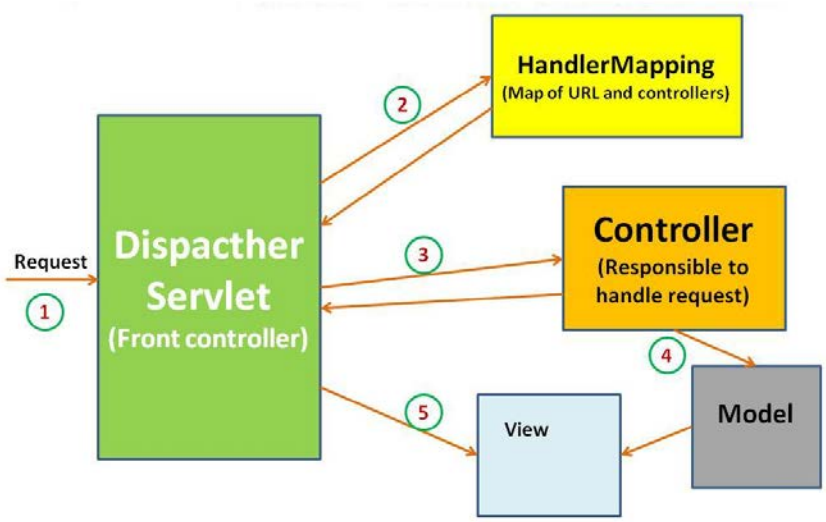


Figura 12 - Comportamento di un Front Controller.

14.5. DOMAIN OBJECT

E' un pattern architetturale che propone una soluzione al problema comune di decomporre un intero sistema, o di un singolo elemento architetturale. Le parti che formano un sistema software infatti, sono spesso caratterizzate da collaborazioni e relazioni di contenimento molteplici e variegate; realizzare queste funzionalità correlate senza cura può portare a un progetto con un'elevata complessità strutturale. La separazione degli interessi è una qualità fondamentale del software, ed è pertanto necessario decomporre il sistema software in modo opportuno.

Il pattern Domain Object incapsula ciascuna responsabilità funzionale distinta di un'applicazione in un elemento auto-contenuto, un oggetto di dominio. Ciascun elemento deve avere un'interfaccia separata dalla sua implementazione, le collaborazioni tra gli elementi devono avvenire solo sulla base della loro interfaccia, e ciascun elemento deve essere internamente coeso e debolmente accoppiato agli altri elementi.

Per fare un esempio del sistema Ritmare, sarà necessario implementare un oggetto di dominio Missione, che incapsulerà i dati relativi ad una missione ed implementerà i metodi con cui quest'oggetto interagirà con gli altri oggetti di dominio. Le responsabilità dell'oggetto riguarderanno solo i dati della missione, e non andranno a modificare i dati dei singoli sensori, o quelli delle misure, che saranno invece gestiti dai relativi oggetti di dominio.

Grazie a questo pattern, si rende più agevole un'eventuale modifica od aggiornamento della logica del sistema, così come il riuso degli stessi oggetti di dominio per sistemi software differenti.

14.6. DATA ACCESS OBJECT

E' un pattern architetturale in cui gli oggetti DAO sono componenti software che forniscono un'interfaccia verso un database o un qualche altro meccanismo di persistenza dei dati. Mappando le operazioni dello strato logico con quelle dello strato di persistenza, i DAO forniscono metodi specifici sui dati nascondendo i

dettagli interni del database. Essi separano quindi le operazioni e le informazioni provenienti dai dati gestite dal sistema, dal modo specifico con cui questi dati vengono selezionati e recuperati.

Nel caso di accesso ad un database relazionale classico, vi saranno tanti oggetti DAO quante sono le tabelle nel database, in modo che ognuno di essi implementi le operazioni necessarie al sistema per quella specifica parte di dati. Ad esempio, nel sistema RITMARE, avremo un oggetto 'SensoreDAO' che fornirà allo strato logico le operazioni di accesso ai dati contenuti nella tabella Sensore del database, così come avremo un oggetto 'CanaleDAO', un oggetto 'MisuraDAO' eccetera.

14.7. ARCHITETTURA FINALE DEL SISTEMA

Dopo aver analizzato alcuni dei pattern architetturali più utilizzati nella progettazione di sistemi informativi su web, andiamo a definire l'architettura finale scelta per il sistema, che si basa prevalentemente sui modelli descritti in precedenza.

Possiamo inoltre iniziare ad introdurre le scelte a livello di sviluppo software e di linguaggi di programmazione con cui è stato implementato il sistema:

- lo strato di presentazione e di interfaccia utente è stato realizzato mediante l'uso di pagine web, quindi in linguaggio **HTML**, con l'ausilio di fogli di stile **CSS** e di metodi **Javascript**. Le pagine che richiedono chiamate al server sono programmate in **JSP (Java Server Pages)** con alcune estensioni della libreria **JSTL (JavaServer Pages Standard Tag Library)**
- per lo strato di logica si è reso necessario usare un linguaggio di programmazione ad oggetti, e la scelta è ricaduta su **Java**. In particolare, trattandosi di un sistema su web, si è utilizzato il framework Java open-source **Spring**, uno dei più usati in ambito di programmazione di piattaforme web, che fornisce una solida architettura e decine di estensioni per le operazioni più comuni in questo ambito. Spring inoltre fa uso, soprattutto a livello di gestione della configurazione, di file **XML**
- per lo strato di persistenza, abbiamo un file-system in cui i dati grezzi vengono immagazzinati in formato **JSON**, e parallelamente, un Database relazionale (si è scelto anche qui una piattaforma open-source come **MySQL**) che sarà quindi interrogato utilizzando il linguaggio **SQL**
- per quanto riguarda il rilascio del sistema, esso sarà installato su una macchina dedicata, e fatto girare all'interno di un server **Apache Tomcat**, un ambiente specifico in cui far girare applicazioni web basate sul linguaggio Java

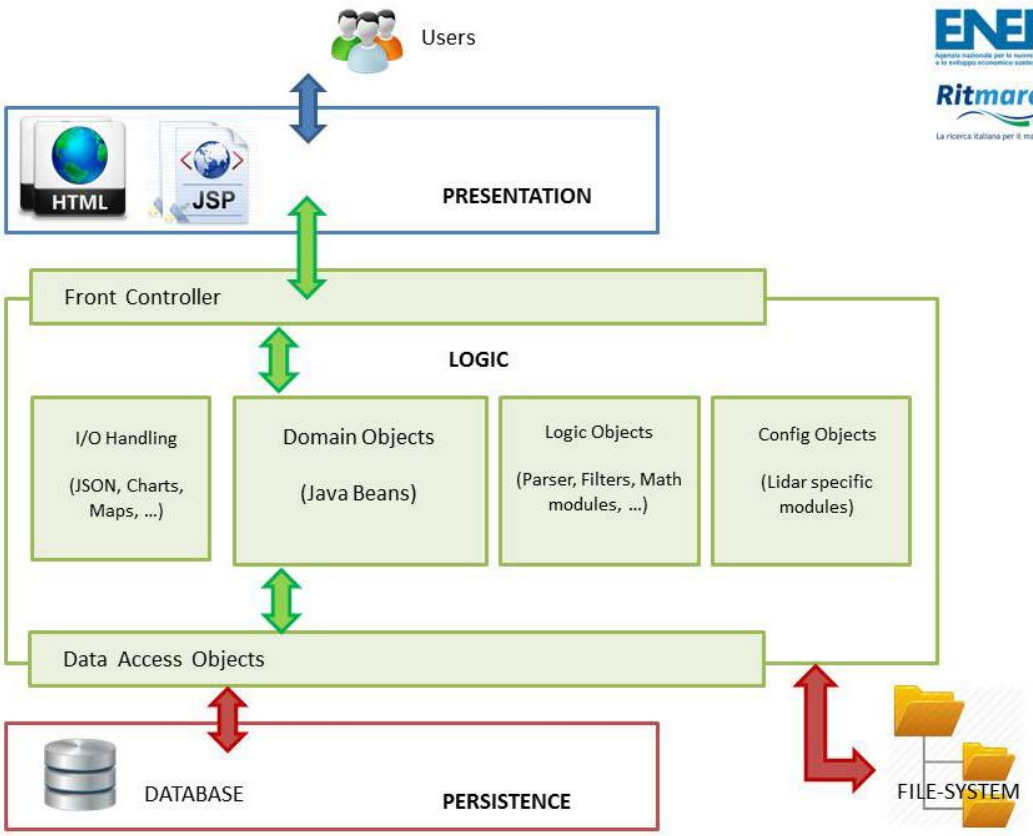


Figura 13 – Architettura finale per il sistema informativo.

Nella Figura 13 è mostrata l'architettura finale scelta per il sistema. In particolare, è possibile notare i tre strati dell'architettura 3-tier e le loro interazioni. Lo strato intermedio di logica è sicuramente quello più ampio, ed è possibile notare in figura alcuni componenti raggruppati per funzionalità:

- gli oggetti di dominio (che in Java sono implementati mediante le classi Bean) e che corrispondono idealmente a oggetti del mondo reale con funzioni distinte
- i componenti che si occupano della logica vera e propria del sistema, ad esempio i parser dei file dati, i filtri da applicare sulle misure eccetera
- i componenti che si occupano della gestione dell'input/output, ad esempio generando grafici, mappe o elaborando stringhe JSON
- i componenti che vanno ad interagire con la configurazione specifica del sensore LIDAR, ad esempio lanciando una shell che va a dialogare con il modulo VEGA di cui abbiamo parlato precedentemente

15. DIAGRAMMA DEI PACKAGE

15.1. DIAGRAMMA DEI PACKAGE

Nella Figura 14 sono descritti i principali package dell'applicazione, con le loro funzionalità. Per package intendiamo un gruppo di classi (in questo caso di classi Java) riunite per motivi concettuali o funzionali.

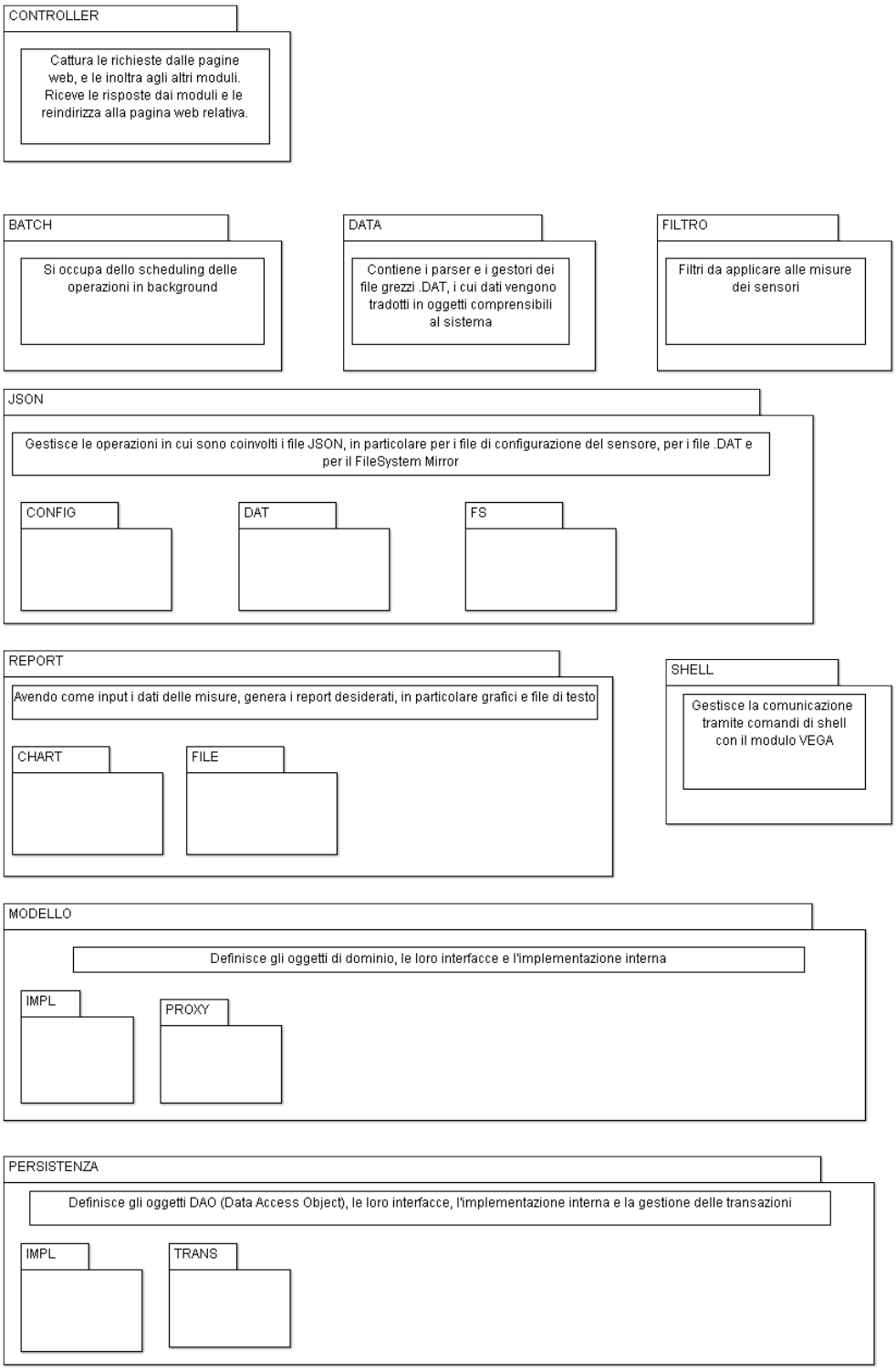


Figura 14 – Package dell'applicazione e loro funzionalità.

Nella Figura 15 è invece mostrato l'albero della struttura dei package e delle directory del sistema, così come appare nell'IDE (Integrated Development Environment) **Eclipse**, l'applicazione utilizzata nello sviluppo del codice del progetto.

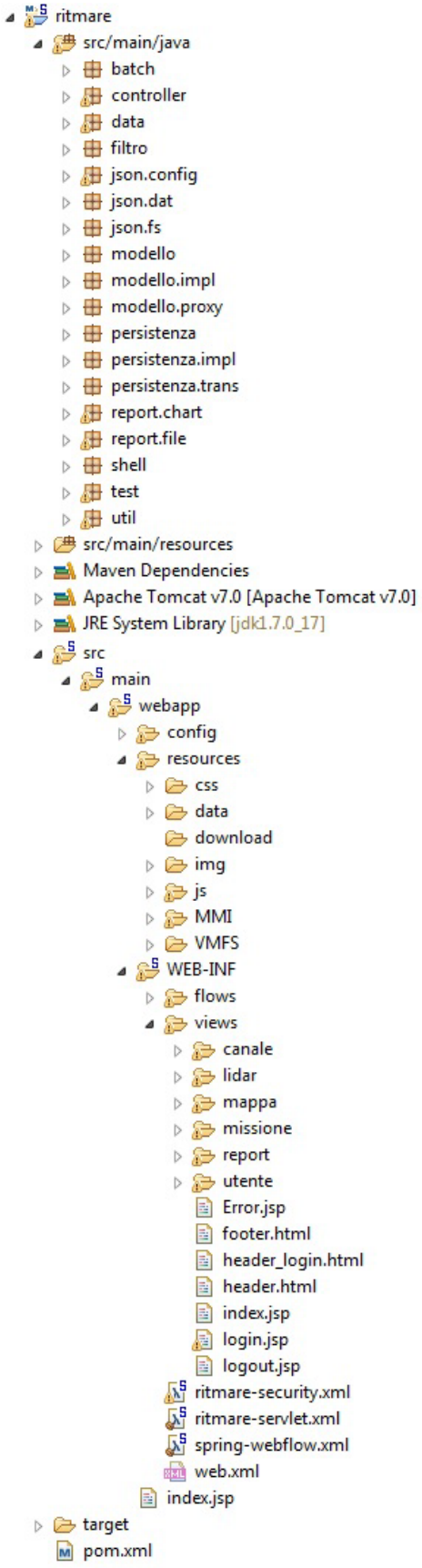


Figura 15 - Albero della struttura dei package e delle directory del sistema

Si può notare che nella Figura 15, l'elenco dei package, già descritto precedentemente, ha a che fare con lo strato logico dell'applicazione. A partire dalla cartella **SRC**, sono elencati i packages appartenenti allo strato di presentazione.

In particolare le due directory più importanti sono **WEB-INF**, che contiene l'elenco completo delle pagine web (HTML e JSP) usate dall'applicazione e strutturate in sotto-cartelle per semplicità, e la directory **RESOURCES**.

Quest'ultima contiene file di diversa natura gestiti dal sistema per scopi diversi. Per esempio:

- La cartella **CSS** contiene i fogli di stile delle pagine web
- La cartella **DATA** contiene file dati temporanei usati per generare report
- La cartella **DOWNLOAD** contiene (temporaneamente) i file che gli utenti hanno confermato di voler scaricare
- La cartella **IMG** contiene le immagini e le icone usate dal sistema
- La cartella **JS** contiene i file Javascript sfruttati nelle pagine web
- La cartella **MMI** contiene il modulo VEGA per la comunicazione con il sensore
- La cartella **VMFS** contiene il file-system con i file .DAT e i file di configurazione del sensore

16. DIAGRAMMA DI SEQUENZA

16.1. DIAGRAMMA DI SEQUENZA (UC1 – Visualizza Report Dati Lidar Grezzi)

Nella Figura 16 sarà rappresentato il flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC1; ossia nel momento in cui l'utente ha sottomesso una form HTML per la visualizzazione di un report, e si aspetta una pagina web di risposta contenente i grafici delle misure lidar, in base ai parametri da lui scelti.

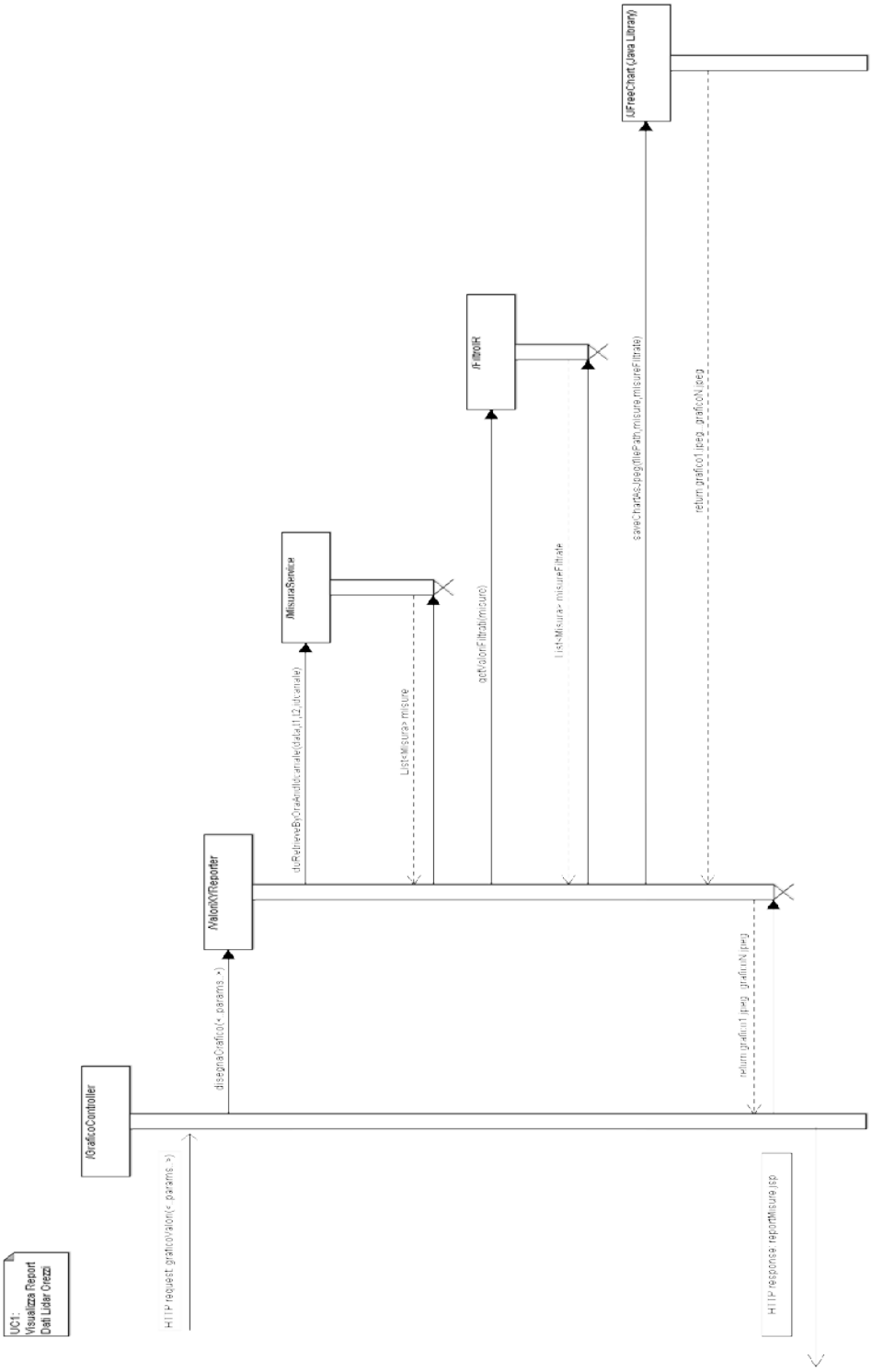


Figura 16 -Flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC1

1

16.2. DIAGRAMMA DI SEQUENZA (UC2 – Visualizza Report Dati Lidar Pre-elaborati)

Nella Figura 17 sarà rappresentato il flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC2; ossia nel momento in cui l'utente ha sottomesso una form HTML per la visualizzazione di un

report e si aspetta una pagina web di risposta contenente i grafici delle misure lidar, in base ai parametri da lui scelti.

In particolare, nel diagramma seguente è riportato il caso di un report che calcola l'errore tramite il metodo della deviazione standard del rapporto fra i valori di un certo canale e quelli di un canale fisso di riferimento. Il tipo di grafico scelto dall'utente è del tipo chiamato 'Line' (l'altro tipo è 'Box').

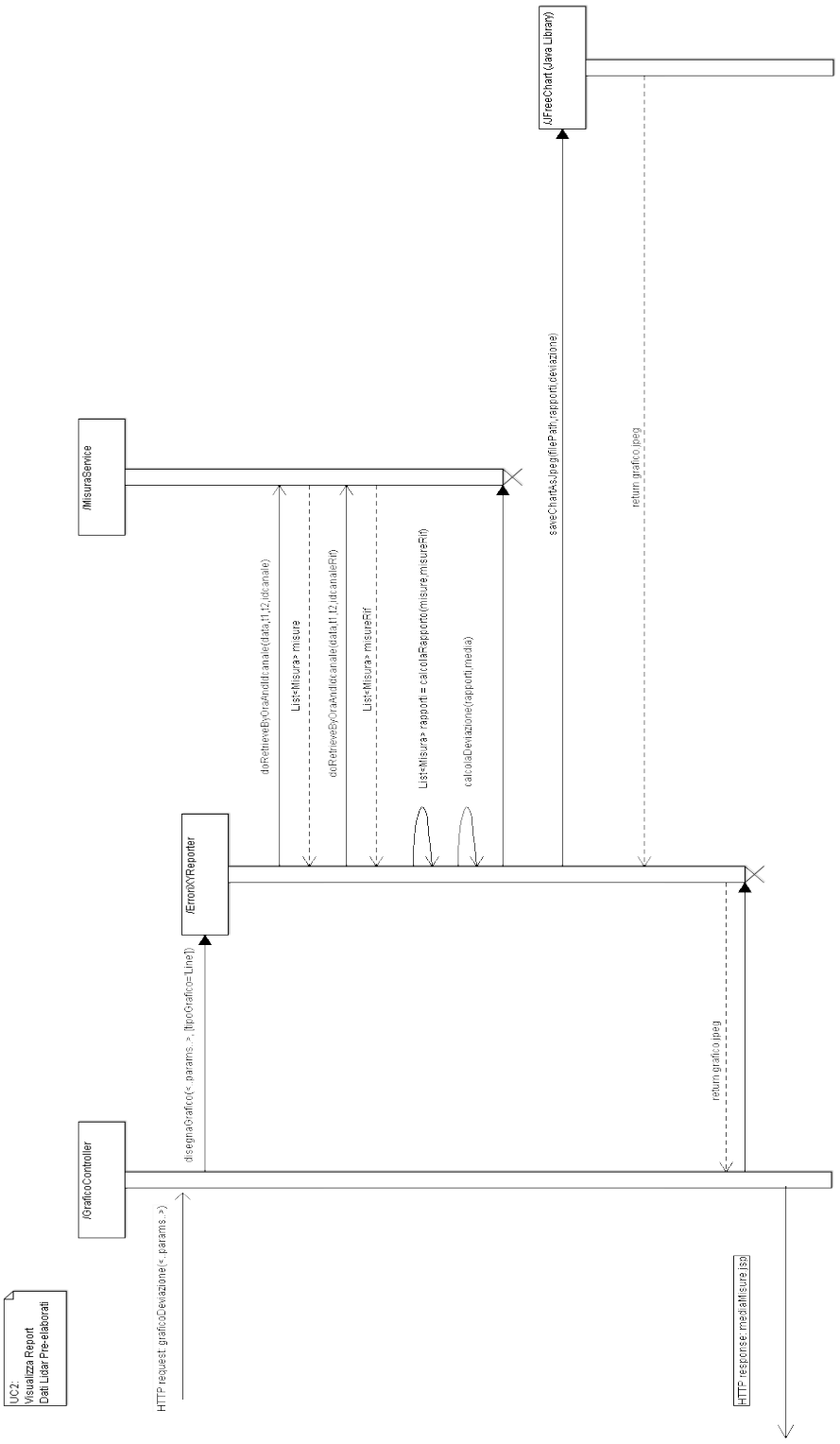


Figura 17 - Flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC2

16.3. DIAGRAMMA DI SEQUENZA (UC3 – Visualizza Dati di Navigazione)

Nella Figura 18 sarà rappresentato il flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC3; ossia nel momento in cui l'utente ha sottomesso una form HTML per la visualizzazione di una mappa contenente la traccia del percorso di navigazione, colorata in base ad un parametro scelto. Il caso mostrato fa riferimento ad una visualizzazione della mappa on-line, che sfrutta le librerie pubbliche di Google Maps.

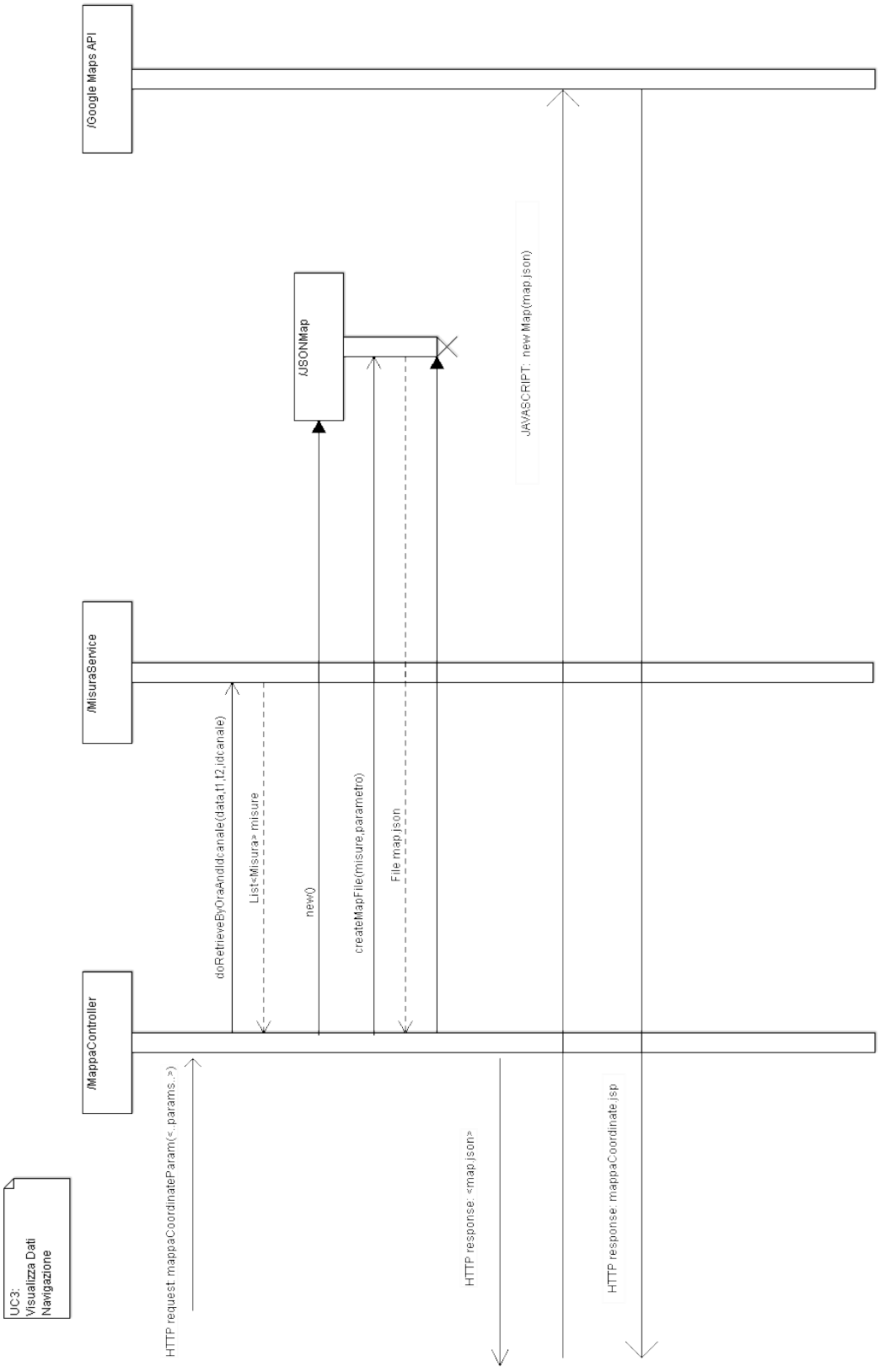


Figura 18 - Flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC3.

16.4. DIAGRAMMA DI SEQUENZA (UC4 – Visualizza Scheda Tecnica del Sensore)

Nella Figura 19 sarà rappresentato il flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC4; ossia nel momento in cui l'utente ha cliccato sulla voce di menu "Scheda Tecnica" con l'obiettivo di visualizzare una pagina web che contiene le specifiche tecniche del sensore. Si assume che l'utente abbia già selezionato la missione e il sensore lidar di interesse.

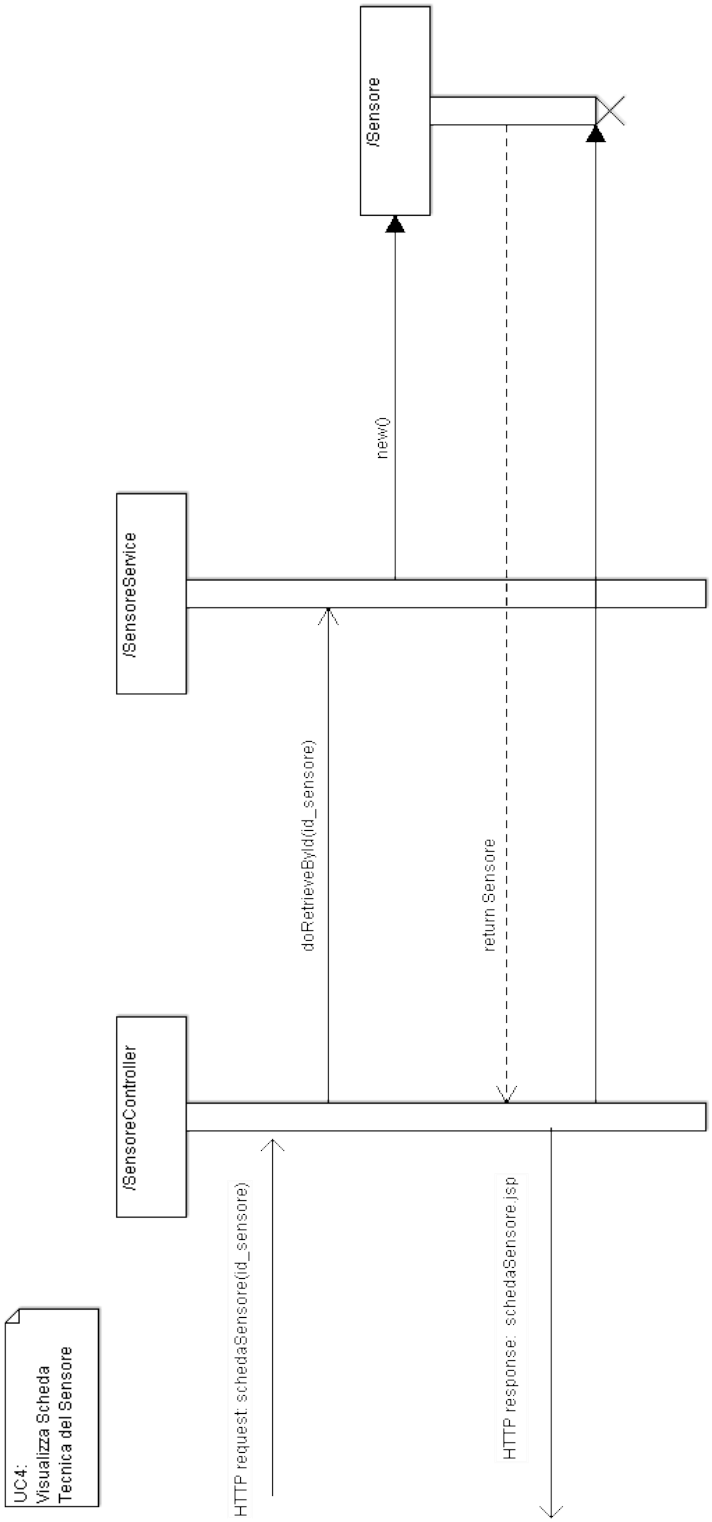


Figura 19 - Flusso di operazioni del sistema nel momento dell'attivazione del caso d'uso UC3.

17. MODELLO DEI DATI

17.1. STRUTTURA DEL FILE DI CONFIGURAZIONE GENERALE config.jsn

Il file di configurazione *config.jsn* racchiude le informazioni generali che descrivono le caratteristiche tecniche del sensore e dei canali relativi. E' presente nella scheda del dispositivo e può essere modificato in alcune delle sue parti dagli operatori per ricalibrare eventualmente i parametri del sensore in corso di missione.

Questo file riveste un'importanza fondamentale in quasi tutte le operazioni svolte dal sistema informativo, in quanto possiede al suo interno i parametri tecnici utilizzati in molte operazioni di recupero, modifica, analisi e salvataggio dei dati. Il file è in formato JSON ed è strutturato in maniera gerarchica secondo i seguenti concetti:

- **MISSIONE**, nodo radice che contiene la descrizione dei parametri della missione
    - **SENSORI**, lista che contiene oggetti Sensore con le relative caratteristiche tecniche (N.B. inizialmente ci sarà sempre un solo sensore per missione, ma si è previsto di implementare una lista per sviluppi futuri in cui saranno presenti più sensori contemporaneamente per missione)
        - **CANALI**, lista che contiene oggetti Canale con le relative caratteristiche tecniche

Una possibile implementazione del file *config.jsn* può essere la seguente:

Nodo Missione:


```
{
   "Missione": {
      "descrizione":"Analisi Mar Tirreno",
      "vettore":"M/N Urania",
      "base_name":"STR1",
      "nome":"Santa Teresa 1",
      "idMissione":1,
      "ente":"ENEA",
      "sensori":[
         {…}
      ]
   }
}
```


Nodi Sensore:

```
"sensori": [
    {
        "idSensore":1,
        "nome":"Lidar LIF",
        "tipo":"LIF",
        "sorgente": {
            "durata_ns":7,
            "energia_mJ":20
        },
        "gate": {
            "tempoIntegrazione_ns":300
        },
        "numeroCanali":4,
        "canali": [2]
    }
]
```


Nodi Canale:


```
"canali": [
    {
        "idCanale":1,
        "larghezza_nm":15,
        "tensioneLavoro_v":422,
        "codice":"F404",
        "curvaCalibrazione":"tipo-1",
        "centroBanda_nm":405,
        "stato":"attivo"
    },
    {
        "idCanale":2,
        "larghezza_nm":15,
        "tensioneLavoro_v":422,
        "codice":"F680",
        "curvaCalibrazione":"tipo-1",
        "centroBanda_nm":680,
        "stato":"attivo"
    }
]
```


17.2. FILESYSTEM DEL DISPOSITIVO LIDAR

Ogni dispositivo LIDAR è dotato di una memoria interna contenente il firmware e una quantità di spazio libero su disco, utilizzato principalmente per salvare le opzioni di configurazione relative al LIDAR stesso, e per memorizzare i dati raccolti durante la missione.

Il *filesystem* presente nella scheda di memoria ha una struttura fissa e un *namespace* delle directory e dei file che risponde a regole precise, in modo da poter essere accessibile e modificabile in modo autonomo e automatico dal server locale collegato al dispositivo. In questo modo, il server può recuperare i dati e le informazioni di cui ha bisogno sapendo già in quale percorso saranno presenti, velocizzando così non solo eventuali richieste da parte degli utenti, ma in particolar modo l'esecuzione di *scheduled tasks* ad orari programmati, decisi a priori dagli operatori.

La radice del *filesystem* deve contenere obbligatoriamente un file di configurazione in formato JSON, che chiameremo **config.jsn**.

Questo file, tra le altre informazioni, raccoglie al suo interno i dati che definiscono la missione in corso (la descrizione completa è spiegata nel paragrafo precedente), in particolare:

- Il ***base-name***, ossia il nome in codice della missione, tipicamente di 4 caratteri (supponiamo ad esempio AX08)
- Il **numero di canali attivi** del dispositivo
- Le specifiche tecniche dei canali

Una possibile configurazione specificata nel file potrebbe essere la seguente:

```
{
“base-name”: “AX08”,
“num_canali”: 4,
“canali”: [{
        “codice” : CH405, …
    },
    …,
    {
        “codice” : CH680, …
    }],
…
}
```

In questo caso, la struttura del *filesystem* sarà modellata in base alle specifiche tecniche contenute nel file di configurazione *config.jsn*. La radice del *filesystem* avrà quindi una directory chiamata come il *base-name* della missione (**\AX08**). All’interno di essa, sarà presente sicuramente una cartella **\DATA** in cui al suo interno sono memorizzati i file generati dal dispositivo con i valori raccolti ora per ora, più altre cartelle accessorie che possono contenere ulteriori file di configurazione, dettagli della missione, e altre informazioni di qualsiasi genere.

Nella cartella DATA, i file in cui sono registrati i valori raccolti dal LIDAR saranno a loro volta suddivisi in sottocartelle chiamate **SECTn**, dove SECT sta per *Section*, ed *n* è un numero progressivo a due cifre. Ognuna di queste cartelle dovrà contenere quindi da un minimo di 1 ad un massimo di 100 file di dati, ordinati anch’essi progressivamente. Per esempio, nel caso in cui il LIDAR raccolga 250 file dati, essi verranno suddivisi nelle seguenti cartelle:

- **..\DATA\SECT01** con i primi 100 file

- **..\DATA\SECT02** con i successivi 100 file
- **..\DATA\SECT03** con gli ultimi 50 file

I file contenenti i dati del LIDAR (caratterizzati da un'estensione **.DAT**) avranno un nome ricavato dal *base-name* della missione e da un numero progressivo di 4 cifre stabilito al momento del salvataggio: **BBBBnnnn.DAT**. Supponendo che una missione di *base-name* STTR duri 48 ore, e che le misurazioni vengano salvate una volta all'ora, ci saranno 48 file .DAT a rappresentare i dati raccolti, dal file **STTR0000.DAT** al file **STTR0047.DAT**, tutti salvati nella cartella SECT01.

Riassumendo, in un'ipotetica missione di base-name AX08, che abbia raccolto dati ogni ora per 200 ore, la struttura del *filesystem* nella memoria del dispositivo sarà la seguente:

***config.jsn**	**file di configurazione generale**
\AX08\CONFIG\...	**directory con ulteriori dati di configurazione, o backup di precedenti configurazioni**
\AX08\EXTRA\...	**directory con informazioni supplementari della missione**
\AX08\DATA\SECT01*AX080000.DAT*	
…	**directory con i primi 100 file dati**
\AX08\DATA\SECT01*AX080099.DAT*	
\AX08\DATA\SECT02*AX080100.DAT*	
…	**directory con i successivi (e ultimi) 100 file dati**
\AX08\DATA\SECT02*AX080199.DAT*	

17.3. STRUTTURA DEI FILE DATI (.DAT) MEMORIZZATI SUL DISPOSITIVO

Ad intervalli di tempo prefissati, il dispositivo LIDAR salverà i dati raccolti nell'ultima finestra temporale, memorizzandoli su un file di estensione DAT. Abbiamo già detto che il nome del file sarà generato in base al codice della missione e ad un numero progressivo che indica la finestra temporale (per esempio, AX0080001.DAT). Vediamo dunque come saranno strutturati i dati all'interno di questi file.

Come premessa, stabiliamo che la struttura di questi file dati dovrà essere dinamica: ogni file infatti, potrà avere più o meno campi in base alla configurazione con cui il LIDAR sta operando. Per esempio, un file potrà contenere i dati raccolti da 4 canali diversi, mentre un altro solo da 2 canali. Oppure un file potrà contenere informazioni temporali scritte in formato GPS piuttosto che in altri formati.

Per aggirare le difficoltà di elaborare in maniera automatica file di struttura differente, aggiungeremo come *header* del file (nella realtà semplicemente la prima riga di testo) una stringa in formato JSON che indicherà l'ordine e i campi contenuti nelle righe successive. In questo modo, il server saprà già in anticipo che tipo di file andrà a processare ed a memorizzare nella sua base di dati, per la consultazione da parte degli utenti.

Un esempio di stringa JSON che rappresenta *l'header* di un file .dat in cui sono memorizzati i tempi, le coordinate, e i valori raccolti da due canali potrebbe essere la seguente:

{ "header": [
{ "basename": "COSIMO" }, { "date": "2015-03-28" }, { "time": "09:07:09" }, { "gps_time": "N/A" },
{ "gps_lat_dm": 0 }, { "gps_lon_dm": 0 }, { "gps_heading": 0 }, { "gps_speed": 0 },
{ "aux_radiance_uMol": 0 }, { "aux_extTemp_deg": 19.31 }, { "aux_press_mBar": 0 },
{ "aux_pulse_mJ": 0 }, { "aux_ch": 0 }, { "src_sts": "Q" }, { "src_energy": 4 }, { "acq_mode": "F" },
{ "acq_avg": 32 }, { "ch1_id": 405 }, { "F0": 248.7638 }, { "R0": 0 }, { "B0": 0 }, { "T0": 20.31 },
{ "P0": 410.31 }, { "ch2_id": 450 }, …dati di altri canali …] }

Da notare che il codice del canale non è scritto in questo header: F0 o F1 infatti non indicano il canale di codice 0 o 1, ma sono semplicemente gli indici dell'array dei canali presente nel file di configurazione generale *config.jsn*. Il server, avendo già elaborato quel file, sa in anticipo che il canale di indice 0 sarà, per esempio, il 405, e quello di indice 1 il 450.

Il contenuto del file, a partire dalla seconda riga, sarà quindi un insieme di valori separati da una tabulazione (/t). Il numero di valori presenti su ogni riga, sarà chiaramente lo stesso del numero di parametri dichiarati nell'header. Riprendendo l'esempio precedente:

COSIMO 2015-03-28 19:00:09 19:00:08
37.91563333 16.03186667 0.0 0.0
0.0 18.3 0.0
0.0 0 Q 4.3 F
32 405 67.7467 0 0 19.4
400.36 450 …valori altri canali…

Abbiamo già stabilito che i file saranno memorizzati ad intervalli di tempo prefissati. Quindi, ipotizzando che un file venga salvato ogni ora, e che i valori vengano raccolti una volta al secondo, il file di esempio AX080001.DAT avrà 1 riga di header e, al massimo, 3600 righe dati (60 sec * 60 min).

17.4. STRUTTURA DELLA BASE DI DATI DEL WEB SERVER

Una volta che i file di dati grezzi provenienti dal sensore sono stati acquisiti dal sistema, essi saranno strutturati in una base di dati per essere gestiti al meglio dal sistema informativo e per permettere all'utente di compiere analisi e ricerche di diversa natura.

Il modello logico di progettazione della base di dati usata in questo progetto è quello **relazionale**, ossia il modello classico basato sulla logica del primo ordine, in cui tutti i dati sono organizzati in tuple appartenenti a tabelle collegate eventualmente fra loro tramite relazioni.

Nella Figura 20 è mostrato il diagramma EER (*Enhanced Entity-Relationship*) della base di dati, dove ogni riquadro (corrispondente ad una tabella) contiene l'elenco degli attributi (campi), e le frecce indicano relazioni (tramite *foreign keys*) tra una tabella e l'altra. Accanto ad ogni attributo è indicato il tipo di dato corrispondente.

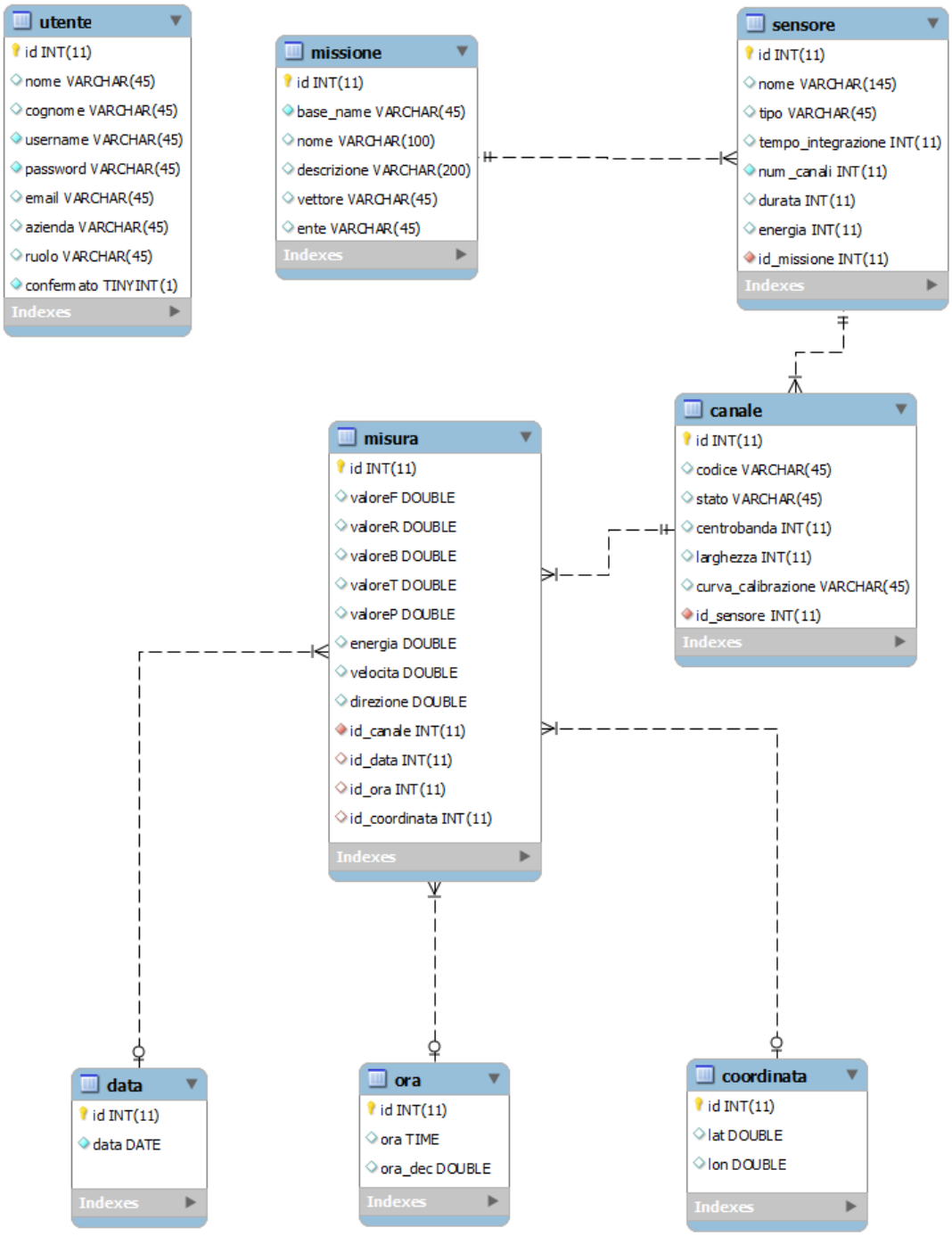


Figura 20 – Diagramma Entità – Relazione della base di dati.

17.5. ANALISI VALORI FLUORESCENZA

Una delle operazioni principali che il sistema permette di compiere è quella di visualizzare su un grafico x-y l'andamento dei valori di fluorescenza di uno o più canali in un determinato intervallo di tempo scelto dall'utente. Nel grafico saranno presenti, per ogni canale, dei piccoli cerchi (outlier) corrispondenti ai valori effettivi, ed una linea continua che corrisponderà ai valori attuali filtrati per mezzo di un filtro IR.

L'algoritmo utilizzato per il filtro è il seguente:

```
N =10;                                  //valore costante
Per ogni valore f della lista Misure:
    IF (f è il primo valore della lista) THEN
        ff ← (f/N) + (f*(N-1/N))
    ELSE
        ff ← (f/N) + (i * (N-1/N))
    i ← ff                              //variabile d'appoggio per l'iterazione
    lista MisureFiltrate.add(ff)
```

Nella Figura 21 si può vedere un esempio degli output di questa analisi:

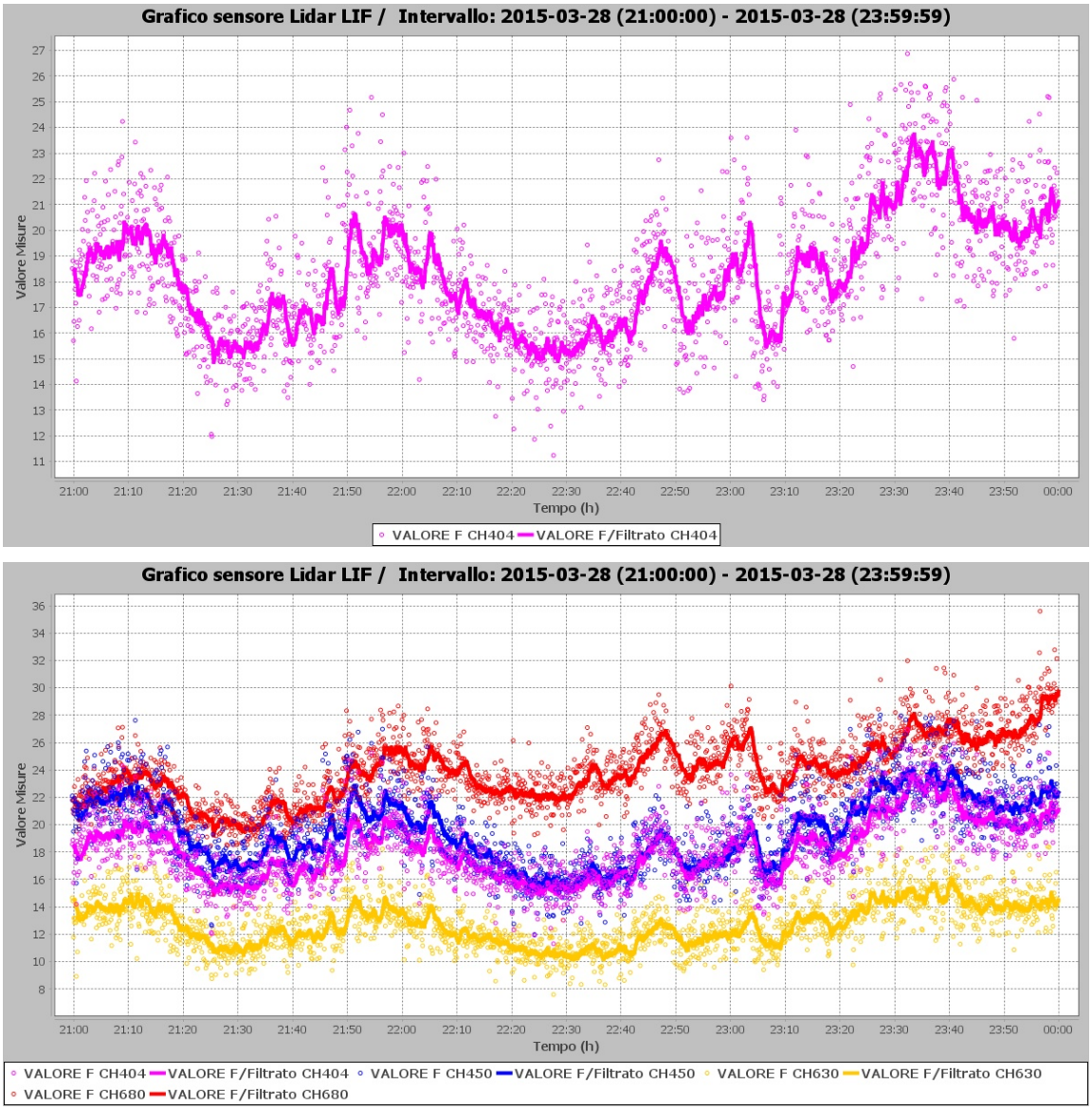


Figura 21 – Segnali registrati con il Lidar fluorosensore nella campagna del 28/03/2015 in Adriatico per il canale Raman @ 404 nm (sopra) ed il confronto con tutti gli altri canali acquisiti (sotto).

18. ANALISI RAPPORTO VALORI FLUORESCENZA

Una delle analisi che permette di avere dati LIDAR pre-elaborati è quella del rapporto dei valori di fluorescenza tra un canale scelto dall'utente e un canale fisso di riferimento (in questo caso il 404). Questi rapporti fra valori saranno mediati in un intervallo a scelta. Inoltre, verrà calcolato lo scarto di errore tramite il metodo della deviazione standard.

L'algoritmo utilizzato per questa analisi è il seguente:

Per ogni intervallo temporale di media int_media:

listaRif ← valori del canale di riferimento

listaVal ← valori del canale selezionato

IF (liste di dimensioni diverse) THEN ERROR

mediaRif ← media dei valori del canale di riferimento

mediaVal ← media dei valori del canale selezionato

rapporto ← mediaVal/mediaRif //rapporto delle medie (è stato scelto rispetto al metodo della media dei rapporti)

*/*Calcolo deviazione standard*/*

INV ← 1/listaVal.size() //inverso della dimensione della lista

SOMMATORIA <- (mediaVal – listaVal[k])² per ogni valore k di ListaVal

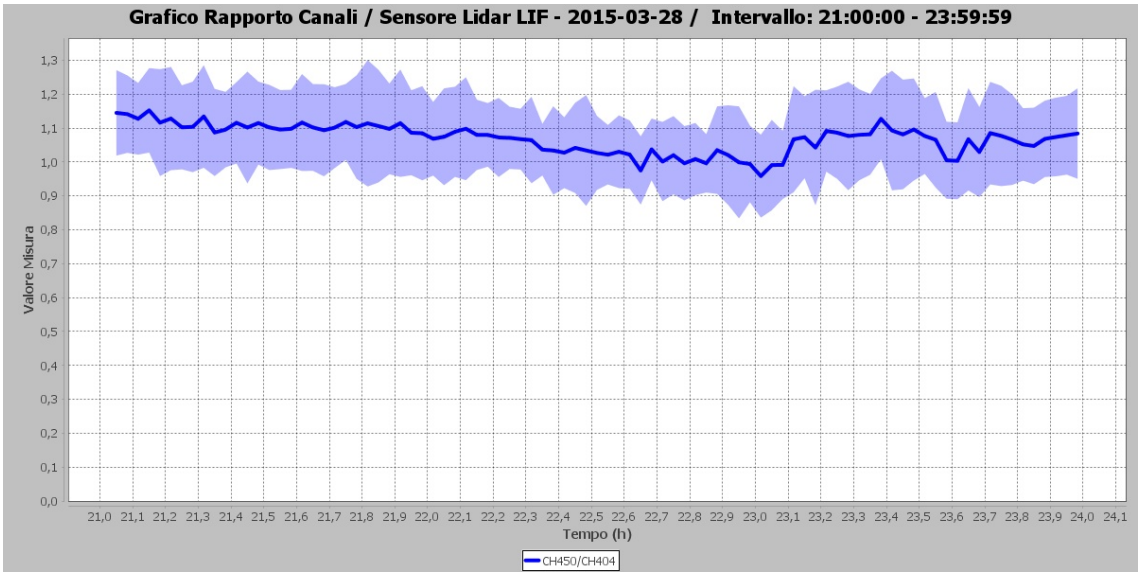
$DEV\_Val \leftarrow \sqrt{INV * SOMMATORIA}$

DEV_Rif ← ..stesso metodo con valori di ListaRif

$$ERR \leftarrow \sqrt{\left(\frac{DEV\_Val}{mediaRif}\right)^2 + \left(\frac{mediaVal * DEV\_Rif}{mediaRif^2}\right)^2}$$

listaFinale.add(rapporto,rapporto+ERR,rapporto-ERR)

Nella Figura 22 si può vedere un esempio degli output di questa analisi (tempo di media 2 minuti):



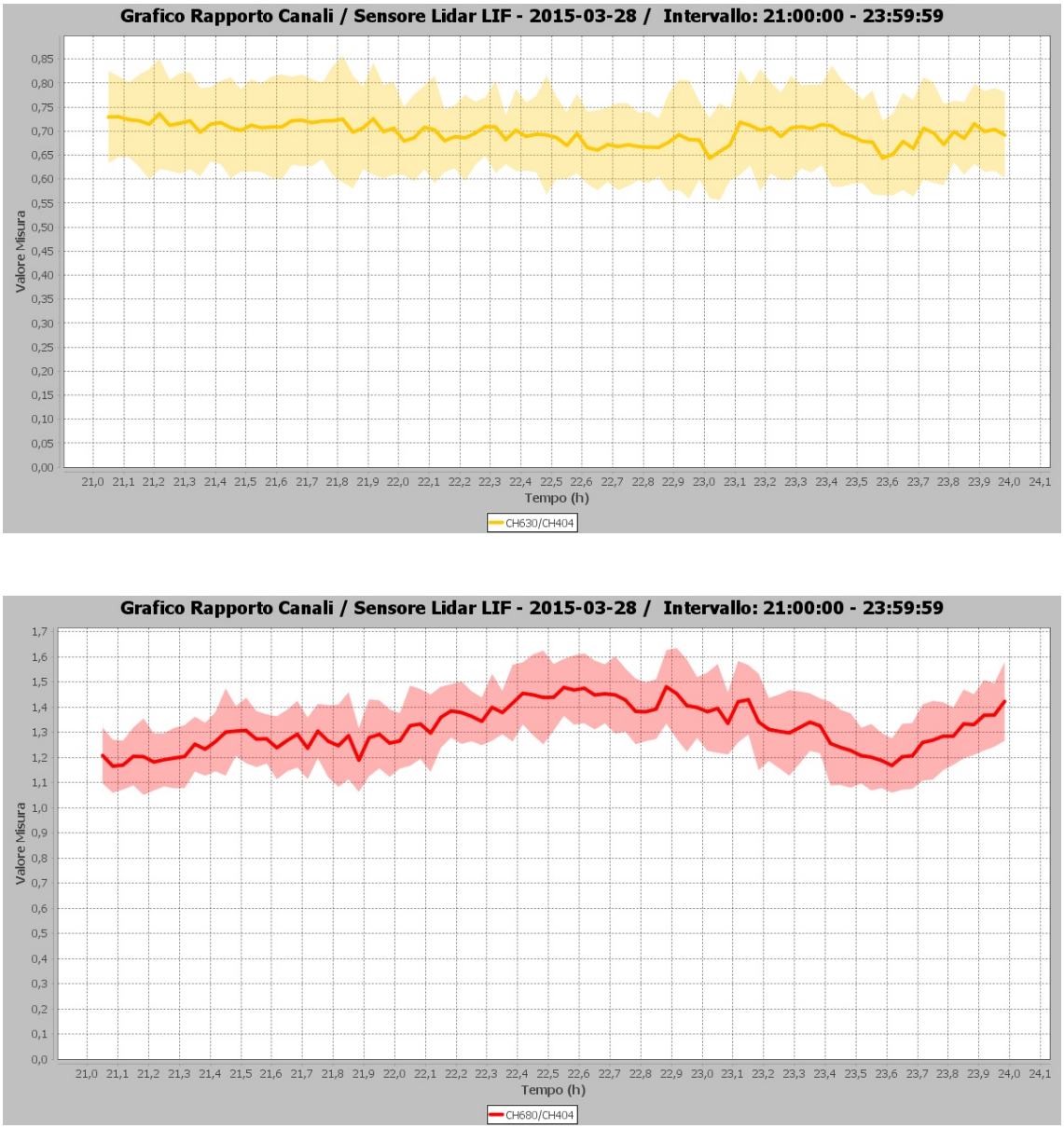


Figura 22 – Canali spettrali misurati dal Lidar fluorosensore nella campagna del 28/03/2015 in Adriatico normalizzati al rispettivo canale Raman @ 404 nm; Canale del DOM @450 nm (sopra); Canale Ficoeritrina (nel mezzo) e la Clorofilla @680 nm (sotto).

19.BIBLIOGRAFIA

1 Comitato Promotore della Piattaforma Tecnologica Nazionale Marittima, Programma Ritmare: la ricerca italiana per il mare, Ministero Infrastrutture e Trasporti 2010

2 http://www.ritmare.it

3 Parikh, N. C., Parikh, J. A., Clark, M., Damon, M., Mandable, S., & Connors, M., An Extensible System Architecture for Web-Based LIDAR Experimentation and Data Analysis, 22nd Internation Laser Radar Conference (ILRC 2004), Proceedings of the Conference held 12-16 July, 2004 in Matera, Italy. Edited by Gelsomina Pappalardo and Aldo Amodeo. ESA SP-561. Paris: European Space Agency, 2004, p.243

4 Jaeger-Frank, E., Crosby, C. J., Memon, A., Nandigam, V., Conner, J., Arrowsmith, J. R., ... Baru, C. (2006). A three tier architecture applied to LiDAR processing and monitoring. Scientific Programming, 14(3-4), 185-194

5 https://en.wikipedia.org/wiki/Multitier_architecture

6 https://en.wikipedia.org/wiki/Model-view-controller

ENEA
Servizio Promozione e Comunicazione
www.enea.it

Stampa: Laboratorio Tecnografico ENEA - C.R. Frascati
giugno 2017