Progetto IN.TER.PA

CUP D44 E18 000 020 006

Azione A2: Individuazione di tutte le componenti del "kit del riuso" della buona pratica

Ente responsabile del coordinamento dell'Azione: Regione Campania

Indicatore di Output /1				
Azione	Indicatore Unità di misura Valore Target	Unità di misura	Valore Target	Risultato attività
A.2.1.	Documenti di analisi gestionale e organizzativa della buona pratica con quadro economico, cronoprogramma e funzioni	Numero	1	Capitolo 2
A.2.2	Documento di analisi tecnologica della buona pratica	Numero	1	Capitolo 3
A.2.3	Documento di analisi del contesto normativo, regolamentare, e organizzativo contenente la mappatura degli ambiti di applicazione della buona pratica e il work flow	Numero	1	Capitolo 4
A.2.4	Documento sintesi della buona pratica	Numero	1	Capitolo 5
A.2.5	Documento di gestione del trasferimento e adozione della buona pratica	Numero	1	Capitolo 6
A.2.6	Documento di gestione organizzativa del trasferimento della buona pratica	Numero	1	Capitolo 7
A.2.7	Pacchetto documentale di gestione degli aspetti tecnologici per il trasferimento di adozione della buona pratica	Numero	1	Capitolo 8
A.2.8	Pacchetto documentale di gestione degli aspetti amministrativi del trasferimento e gestione della buona pratica	Numero	1	Capitolo 9

Ver.	Elabora	Verifica	Approva	Data emissione	Descrizione delle modifiche
1.0	AlmavivA S.p.A. soggetto attuatore SPC CLOUD LOTTO 4	RT	RT	30/09/2019	Prima stesura del documento
	Barbara Pacileo	Simone Secci	Sauro del Turco		

SOMMARIO

1 INTRODUZIONE

Il Progetto si propone di diffondere nei contesti organizzativi-operativi delle Regioni Toscana e Molise e del Comune di Torella dei Lombardi l'utilizzo del Modello di Gestione Integrata delle Entità e degli Eventi territoriali e del relativo Geographic Cloud I.TER. già realizzato dalla Regione Campania come buona pratica.

La strategia prevede la collaborazione tra i partner per la redazione del KIT DEL RIUSO con l'analisi degli aspetti gestionali/organizzativi/tecnologici/amministrativi/informativi delle azioni di ricerca, selezione, trasferimento, adozione e gestione a regime della buona pratica. Le Strutture competenti in ICT dei Partner collaboreranno alle attività di installazione di I.TER. presso un Data Center condiviso e sua successiva erogazione ai partner in modalità di Cloud Computing. Verranno individuate all'interno dei partner regionali riusanti Direzioni pilota, che successivamente al trasferimento del Know How organizzativo/tecnico e la messa a sistema dei data base di interesse applicheranno il Modello alla gestione di eventi territoriali di natura trasversale e di interesse strategico (Rete Natura 2000, rischio sismico, gestione dei rifiuti, bonifica dei siti inquinati, Piani di Emergenza Ambientali, risorse energetiche, agricoltura, sviluppo rurale, etc). La sfida che si intende affrontare con questa Strategia è quella di creare una rete di cooperazione interregionale, intraregionale e interistituzionale che collabori alla gestione e allo sviluppo della buona pratica apportando concrete evoluzioni al modello originale. L'evoluzione della buona pratica intende diffondere l'uso di Geo-Community/Apps Mobile per favorire i meccanismi di cittadinanza digitale e mettere a punto, anche attraverso il contributo del Comune partner Torella dei Lombardi, una collaborazione Regione /Comuni per il reciproco scambio di dati per affinare la conoscenza del territorio.

2 ANALISI GESTIONALE E ORGANIZZATIVA DELLA BUONA PRATICA CON QUADRO ECONOMICO, CRONOPROGRAMMA E FUNZIONI

2.1 CRONOPROGRAMMA

Si riporta il cronoprogramma di dettaglio relativo ai sottosistemi di i.Ter a partire dalla fase di progettazione fino alla fase di messa in produzione su ambiente cloud

ID	Nome attività	Inizio	Fine
1	Project Management	21/09/15	30/12/16
2	Assessment sistemi di elaborazione	21/09/15	16/10/15
3	Assessment Sistemi Informatici	21/09/15	16/10/15
4	Assessment Banche Dati	21/09/15	16/10/15
5	Piattaforma applicativa per Test e Collaudo	21/09/15	30/10/15
6	Realizzazione I.Ter Data	12/10/15	01/01/16
7	Realizzazione I.Ter Admin	12/10/15	04/12/15
8	Realizzazione I.Ter GIS	12/10/15	04/12/15
9	Realizzazione I.Ter Service – servizi GIS	12/10/15	04/12/15
10	Realizzazione I.Ter SSD	12/10/15	29/01/16
11	Realizzazione Demanio e Patrimonio	12/10/15	01/01/16
12	Realizzazione I.Ter Service	12/10/15	04/12/15
13	Realizzazione I.Ter Portal	12/10/15	01/01/16
14	Integrazione SOA	11/01/16	04/03/16
15	Realizzazione I.Ter Form	11/01/16	04/03/16
16	Formazione - Analisi fabbisogni formativi	16/11/15	04/12/15
17	Formazione - Pianificazione esecutiva e progettazione	03/12/15	23/12/15
18	Formazione - Piattaforma e contenuti	11/01/16	04/03/16
19	Formazione - in alula	07/03/16	29/04/16
20	Servizi di Start-UP	02/05/16	09/12/16
21	Avvio Quinto d'obbligo	01/04/17	15/09/18
22	Realizzazione i.Ter GeoFleet	01/04/17	30/06/17
23	Realizzazione Tab2Geo	01/04/17	30/06/17
24	Realizzazione BlockChain	01/05/17	30/06/17
25	Realizzazione Machine Learning	16/04/17	30/06/17

24	Migrazione su ambiente Cloud i.Ter Base	02/05/18	31/05/18
25	Migrazione su ambiente Cloud i.Ter V	20/06/19	10/09/19

2.2 COSTI

L'importo a base d'asta è stato di €. 3.470.167,00 (oltre IVA) su fondi POR FESR Campania 2007/2013 Obiettivo 5.1. Con D.D. n. 7 del 25/1/2015, la DG 10 ha aggiudicato l'appalto al RTI Almaviva/Planetek/Trilogis per un importo pari a 2.966.125.24 euro al netto dell'IVA.

Con D.D. n. 138 del 29/3/2017 della DG 10, è stata successivamente approvata l'offerta tecnico-economica di Almaviva presentata per l'estensione contrattuale per un costo totale di € 467.106,00 oltre IVA a valere su Fondi POR Campania 2014-2020 dell'Obiettivo Tematico 2.

2.3 CARATTERISTICHE ORGANIZZATIVE

La Stazione Appaltante è stata rappresentata dalla Giunta Regionale della Campania. La procedura di gara è stata affidata all'A.G.C. Demanio e Patrimonio, Settore Provveditorato ed Economato. Il contratto con l'impresa aggiudicataria è stato stipulato e gestito dal Settore 02 "Analisi, progettazione e gestione sistemi informativi" dall'Area Generale di Coordinamento 06 "Ricerca Scientifica, Statistica, Sistemi Informativi ed Informatica". La funzione di Responsabile Unico del Procedimento è stata svolta dal dott. Ferdinando Rodriquez.

A seguire viene riportata la struttura organizzativa di Regione Campania. La piattaforma i.Ter coinvolge tutte le Direzioni Generali e gli Uffici Speciali consentendo la collaborazione nella fase di realizzazione e condivisione di analisi territoriali.

3 ANALISI TECNOLOGICA DELLA BUONA PRATICA

Il presente capitolo ha lo scopo di descrivere:

- ✓ Piattaforme Hardware (architettura, parametri dimensionali minimi),
- ✓ Piattaforme Software (stack architetturale, interfacce, interoperabilità, indipendenza componenti)
- ✓ Infrastrutture di Rete (schema della rete, protocolli di comunicazione, requisiti minimi) della Regione Campania

3.1 PIATAFORME HARDWARE E SOFTWARE

VM Componente	Nome Macchina	Porta	Area Logica	Tipologia Ambiente	RAM minima richiesta (GB)	Sistema Operativo	vCPU	Hardisk (GB)	Modello Processore	IP privato (attuale)	Software installato	DETAIL SW (versioni)
Back-end	Iter-mqtt1	1883, 8883, 61616 ,8161, 61613	Broker	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.14	ActiveMQ	5.14.5
Back-end	Iter-mqtt2	1883, 8883, 61616 , 8161, 61613	Broker	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.15	ActiveMQ	5.14.5
Back-end	Iter-mqtt3	1883, 8883, 61616 , 8161, 61613	Broker	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.16	ActiveMQ	5.14.5
Back-end	Iter-cep1	9611, 9999, 9711, 40849 , 7611, 7711, 9443, 9763, 11111		Cloud TIM	16	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.17	WSO2CEP	4.2.0
Back-end	Iter-cep2	9611, 9999, 9711, 40849 , 7611, 7711, 9443,		Cloud TIM	16	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.18	WSO2CEP	4.2.0

VM Componente	Nome Macchina	Porta	Area Logica	Tipologia Ambiente	RAM minima richiesta (GB)	Sistema Operativo	vCPU	Hardisk (GB)	Modello Processore	IP privato (attuale)	Software installato	DETAIL SW (versioni)
		9763, 11111										
Back-end	Iter-esb1	41709, 9999, 8243, 8280, 9443, 9763, 11111		Cloud TIM	16	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.19	WSO2ESB	5.0.0
Back-end	Iter-esb2	41709, 9999, 8243, 8280, 9443, 9763, 11111		Cloud TIM	16	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.20	WSO2ESB	5.0.0
Back-end	Iter-sensori1	8080	Dati	Cloud TIM	8	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.21	Nifi	1.9.2
Back-end	Iter-sensori2	8080	Dati	Cloud TIM	8	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.22	Nifi, Redash	1.9.2, 4.0
Back-end	Iter-sensori3	8080	Dati	Cloud TIM	8	CentOS Linux release 7.4.1708	8	100	Intel Core Processor (Broadwell)	10.244.10.23	Nifi	1.9.2
Back-end	Iter-data1	9200, 9300	Dati	Cloud TIM	16	CentOS Linux release 7.4.1708	4	300	Intel Core Processor (Broadwell)	10.244.10.24	Elasticsearch	5.4.1
Back-end	Iter-data2	9200, 9300	Dati	Cloud TIM	16	CentOS Linux release 7.4.1708	4	300	Intel Core Processor (Broadwell)	10.244.10.25	Elasticsearch	5.4.1
Back-end	Iter-data3	9200, 9300	Dati	Cloud TIM	16	CentOS Linux release 7.4.1708	4	300	Intel Core Processor (Broadwell)	10.244.10.26	Elasticsearch	5.4.1
Front-end	Iter-is1	9443, 9999, 41939, 10389, 39479, 10711, 9763, 11111		Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.27	WSO2IS	5.3.0
Back-end	Iter-is2	9443, 9999, 41939, 10389		Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.28	WSO2IS	5.3.0

VM Componente	Nome Macchina	Porta	Area Logica	Tipologia Ambiente	RAM minima richiesta (GB)	Sistema Operativo	vCPU	Hardisk (GB)	Modello Processore	IP privato (attuale)	Software installato	DETAIL SW (versioni)
		, 39479 , 10711 , 9763, 11111										
Back-end	Iter-Iterml1	50070 , 8088, 8080, 9001	Big Data	Cloud TIM	32	CentOS Linux release 7.4.1708	8	500	Intel Core Processor (Broadwell)	10.244.10.29	Big Data	Si veda nota
Back-end	Iter-Iterml2	50070 , 8088, 8080, 9001	Big Data	Cloud TIM	32	CentOS Linux release 7.4.1708	8	500	Intel Core Processor (Broadwell)	10.244.10.30	Big Data	Si veda nota
Back-end	Iter-Iterml3	50070 , 8088, 8080, 9001	Big Data	Cloud TIM	32	CentOS Linux release 7.4.1708	8	500	Intel Core Processor (Broadwell)	10.244.10.31	Big Data	Si veda nota
Back-end	Iter-Iterml4	50070 , 8088, 8080, 9001	Big Data	Cloud TIM	16	CentOS Linux release 7.4.1708	8	500	Intel Core Processor (Broadwell)	10.244.10.32	Big Data	Si veda nota
Back-end	Iter-Iterml5	50070 , 8088, 8080, 9001	Big Data	Cloud TIM	16	CentOS Linux release 7.4.1708	8	500	Intel Core Processor (Broadwell)	10.244.10.33	Big Data	Si veda nota
Back-end	Iter-obu-bc-node-1			Cloud TIM	8	CentOS Linux release 7.4.1708	4	300	Intel Core Processor (Broadwell)	10.244.10.34	BlockChain	
Back-end	Iter-obu-bc-node-2			Cloud TIM	8	CentOS Linux release 7.4.1708	4	300	Intel Core Processor (Broadwell)	10.244.10.35	BlockChain	
Back-end	Iter-iter-bc-node-1			Cloud TIM	8	CentOS Linux release 7.4.1708	4	500	Intel Core Processor (Broadwell)	10.244.10.36	BlockChain	
Front-end	Iter-gfleet-webapp	443	Web Server	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.37	Tomcat, Elasticsearch	8, 6.4
Back-end	Iter-gfleet-service	22		Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.38	Tomcat	8
Back-end	Iter-gfleet-websocket	9292	Websocket	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.39	Java	1.8

VM Componente	Nome Macchina	Porta	Area Logica	Tipologia Ambiente	RAM minima richiesta (GB)	Sistema Operativo	vCPU	Hardisk (GB)	Modello Processore	IP privato (attuale)	Software installato	DETAIL SW (versioni)
Back-end	Iter-gfleet-mqtt	22	Active MQ support	Cloud TIM	8	CentOS Linux release 7.4.1708	4	100	Intel Core Processor (Broadwell)	10.244.10.40	Java	1.8

3.2 Manuale Installazione Active MQ

In questo paragrafo sono descritti i passaggi per l'installazione, la configurazione e la creazione del servizio del software ActiveMQ. ActiveMQ è un message broker scritto in java ed offre un client completo JMS (Java Message Service).

La macchina iter- mqtt1, iter – mqtt2 e iter – mqtt3 prevedono l'installazione dei seguenti prodotti:

- JDK 1.8
- ActiveMQ 5.14.5

3.2.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

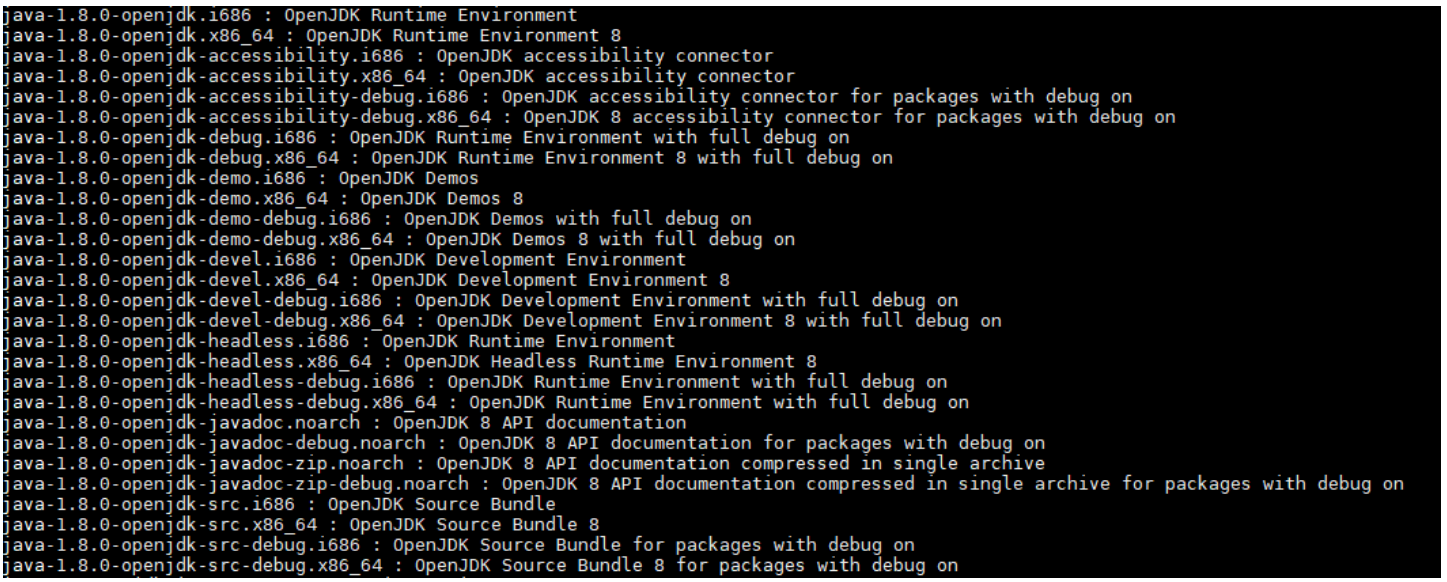
3.2.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:



- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

Per il corretto funzionamento di Tomcat inizializziamo e definiamo i path necessari

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
```

```
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

3.2.3 Installazione ActiveMQ

Per l'implementazione di ActiveMQ è stato utilizzato il file apache-activemq-5.14.5.tar.gz, della macchina corrispondente (in questo caso la iter-mqtt1), il quale contiene la home directory dell'ActiveMQ già esistente con tutte le configurazioni del caso.

- Spostare tramite protocollo SCP il suddetto file sulla macchina da configurare
- Spostare il file appena trasferito nella cartella /opt/:

```
mv apache-activemq-5.14.5.tar.gz /opt/
```

- Scompattare il file

```
tar xfvz apache-activemq-5.14.5.tar.gz
```

- Modificare l'ownership della cartella affidandola all'utente iterusr:

```
chown -R iterusr:iterusr apache-activemq-5.14.5/
```

- Creare il servizio per la gestione del software ActiveMQ

```
vi /etc/systemd/system/activemq.service
```

- Incollare quanto di seguito:

```
[Unit]
Description=HIP Message Broker
After=network.target

[Service]
Type=forking
WorkingDirectory=/opt/apache-activemq-5.14.5/bin
User=iterusr
Group=wheel
ExecStart=/opt/apache-activemq-5.14.5/bin/activemq start
ExecStop=/opt/apache-activemq-5.14.5/bin/activemq stop
Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64

[Install]
WantedBy=multi-user.target
```

- Salvare ed uscire con la sequenza:

```
:wq
```

- Avviare il servizio:

```
systemctl start activemq
```

3.2.4 Monitoraggio

Per controllare che il servizio sia attivo interrogare le porte che il software utilizza per le connessioni:

```
netstat -an | grep 1883
netstat -an | grep 8161
```

Se entrambe sono in stato "LISTEN" allora il servizio è correttamente avviato e attivo.

3.2.5 Comandi per la gestione del servizio

- Avvio del servio:

```
systemctl start activemq
```

- Stop del servizio:

```
systemctl stop activemq
```

- Check del servizio:

```
systemctl status activemq
```

- Restart del servizio

```
systemctl restart activemq
```

- Per il corretto funzionamento, e la corretta comunicazione tra broker e client, in ambiente SMA modificare sulla VM con indirizzo '192.168.56.81' il seguente file:

```
sudo vi
/opt/almaservices/run/confproperties/Wsfleetmanagement/application.yml
```

- Va cambiato quanto evidenziato ed eventualmente l'user e le password:

```
server:
  port: 8900
mqtt:
    host: tcp://10.200.19.185:1883
    user: admin
    password: aCtb9FLqAYVNrEzP
    topicOBU: GEO/OBU/POSITIONS
```

```
    topicAPP: GEO/APP/POSITIONS
    cleanSession: false
    qos: 2
    waitTime: 5000
    timeoutConnection: 300
websocket:
    name: /topic/hi
authentication:
  headertokenkey: it_app_auth
  userviceverifytoken: http://192.168.56.84/dss/session/isValid
  enablecheck: true
```

3.3 Manuale Installazione ITER GFleet

La macchina iter-gfleet-webapp prevede l'installazione dei seguenti prodotti:

- JDK 1.8
- Tomcat 8.5.42
- Elastic Search 6.4.0

La macchina iter-gfleet-service prevede l'installazione dei seguenti prodotti:

- JDK 1.8
- Tomcat 8.5.5

Le macchine iter-gfleet-websocket e iter-gfleet-mqtt prevedono l'installazione della

- JDK 1.8

3.3.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

3.3.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

Per il corretto funzionamento di Tomcat inizializziamo e definiamo i path necessari.

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

3.3.3 Installazione Tomcat

Assicurarsi di aver effettuato l'accesso come utente root.

- Aggiungere gruppo e utente Tomcat per rendere più sicuri gli accessi:

```
groupadd tomcat
useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat
```

- Spostarsi nella directory che ospiterà Tomcat:

```
cd /opt
```

- Assicurarsi che il pacchetto wget sia installato:

```
yum install wget
```

- Effettuare il download di Tomcat dal repository ufficiale:

```
wget http://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.42/bin/apache-tomcat-
8.5.42.tar.gz
```

- Scompattare il file:

```
tar -xzvf apache-tomcat-8.5.42.tar.gz
```

- Rinominare la cartella per facilitarne l'accesso:

```
mv apache-tomcat-8.5.42 tomcat/
```

- Spostarsi ora nella cartella bin di Tomcat per effettuare lo start del prodotto:

```
cd /home/centos/tomcat/bin
```

- Per avviare Tomcat utilizzare il comando seguente:

```
/home/centos/tomcat/bin
```

- I logs di avvio si possono reperire nel file di seguito:

```
cat /home/centos/tomcat/logs/catalina.out
```

- Se la procedura è andata a buon fine, collegandoci dal browser alle macchine ed utilizzando la

porta di default di Tomcat, possiamo vedere la pagina iniziale di benvenuto:

```
http://10.240.10.37:8080
```

Copiare la cartella webapps del vecchio progetto nella nuova directory di Tomcat sostituendo quella di default.

- Modificare i permessi cambiando il proprietario della cartella nell'omonimo user:

```
chown -hR tomcat:tomcat tomcat
```

Creare il servizio di Tomcat in modo da rendere più veloce e semplici il monitoraggio.

- Creare il file:

```
vi /etc/systemd/system/tomcat.service
```

- Inserire all'interno di questi il seguente contenuto:

```
[Unit]
Description=Apache Tomcat Portal Server
After=network.target

[Service]
Type=forking
ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh
User=tomcat
Group=tomcat
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

- Cambiare owner ed assegnare permessi di execute al nuovo service:

```
chmod u+x /etc/systemd/system/tomcat.service
chown tomcat:tomcat /etc/systemd/system/tomcat.service
```

- Utilizzare ora il comando seguente per avviare tomcat:

```
systemctl start tomcat
```

3.3.4 Installazione ElasticSearch

Si proceda ora con l'installazione di ElasticSearch

- Perché funzioni tutto correttamente, creare un nuovo utente, che farà parte di un omonimo gruppo, che sarà specializzato nell'utilizzo di ElasticSearch:

```
groupadd elasticsearch
useradd elasticsearch -d /opt/elasticsearch-6.4.0/
```

- Copiare la cartella di ElasticSearch sotto il seguente path:

```
cp /home/centos/elasticsearch-6.4.0 /opt/ -rf
```

- Assegnare ownership e permessi alla directory appena copiata all'utente creato poco fa:

```
chown -hR elasticsearch:elasticsearch elasticsearch-6.4.0/
```

- Creare il file per il monitoraggio del servizio:

```
vi /etc/systemd/system/elasticsearch.service
```

- Incollare all'interno le seguenti righe:

```
[Unit]
Description=Elasticsearch
Documentation=http://www.elastic.co
Wants=network-online.target
After=network-online.target


[Service]
User=elastic
Group=elastic

ExecStart=/home/elastic/elasticsearch/bin/elasticsearch -p
/home/elastic/elasticsearch/run/elasticsearch.pid --quiet

# StandardOutput is configured to redirect to journalctl since
# some error messages may be logged in standard output before
# elasticsearch logging system is initialized. Elasticsearch
# stores its logs in /var/log/elasticsearch and does not use
# journalctl by default. If you also want to enable journalctl
# logging, you can simply remove the "quiet" option from ExecStart.
StandardOutput=journal
StandardError=inherit

# Specifies the maximum file descriptor number that can be opened by this
process
LimitNOFILE=65536

# Specifies the maximum number of bytes of memory that may be locked into RAM
# Set to "infinity" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml and 'MAX_LOCKED_MEMORY=unlimited' in ${path.env}
#LimitMEMLOCK=infinity

# Disable timeout logic and wait until process is stopped
TimeoutStopSec=0

# SIGTERM signal is used to stop the Java process
KillSignal=SIGTERM
```

```
# Java process is never killed
SendSIGKILL=no

# When a JVM receives a SIGTERM signal it exits with code 143
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

3.3.5 Monitoraggio

- Per controllare che la versione di Java sia quella giusta utilizzare il comando:

```
java -version
```

- Per controllare quale sia il path della JAVA_HOME utilizzare il comando:

```
echo $JAVA_HOME
```

- Verificare che sia uguale alla seguente:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

- Per controllare che il servizio tomcat sia attivo utilizzare il comando di monitoraggio:

```
systemctl status tomcat
netstat -an | grep 8161
```

Se entrambe sono in stato "LISTEN" allora il servizio è correttamente avviato e attivo.

3.3.6 Comandi per la gestione del servizio

- Avvio del servio:

```
systemctl start activemq
```

- Stop del servizio:

```
systemctl stop activemq
```

- Check del servizio:

```
systemctl status activemq
```

- Restart del servizio

```
systemctl restart activemq
```

3.4 INSTALLAZIONE CLUSTER ELASTIC SEARCH

Le macchine iter – data1 e iter – data2 prevedono l'installazione dei seguenti prodotti:

- JDK 1.8
- Elastic Search 5.4.1

3.4.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

3.4.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

3.4.3 Installazione Elastic Search

Per l'implementazione del software sono stati utilizzati le configurazioni dell'ambiente già esistente e migrati sulle macchine corrispondenti (in questo caso iter-data2).

Sono stati esportati nello specifico la home directory e la directory in cui si trovano i file di configurazione di Elastic Search, rispettivamente (a sinistra il path in cui si possono consultare, a destra il nome del file compresso):

```
/usr/share/elasticsearch - elasticsearch-5.4.1.tar.gz
/etc/elasticsearch – etcelastic.tar.gz
```

- Dopo il trasferimento dei file -in formato .tar.gz- tramite protocollo SCP, scompattarli, a turno, e metterli nelle relative cartelle di destinazione:

```
tar xfvz elasticsearch-5.4.1.tar.gz
mv elasticsearch/ /usr/share/
```

- Ed ora l'altro file:

```
tar xfvz etcelastic.tar.gz
mv elasticsearc/ /etc/
```

- Per completare la configurazione Elastic Search necessità di altre cartelle per il proprio funzionamento, utilizzare dunque i seguenti comandi per la creazione di esse:

```
mkdir /var/lib/elasticsearch
mkdir /var/run/elasticsearch
touch /etc/sysconfig/elasticsearch
```

- Elastic Search non può essere avviato da utente root per possibili problemi durante l'esecuzione. Creare dunque un utente addetto all'esecuzione del servizio garantendogli i permessi necessari:

```
groupadd elasticsearch
useradd elasticsearch -g wheel
usermod -aG wheel elasticsearch
```

- Cambiare ownership delle directory create prima:

```
chown root:elasticsearch /etc/elasticsearch/ -R
chown -R elasticsearch:elasticsearch /var/lib/elasticsearch/
chown -R elasticsearch:elasticsearch /var/run/elasticsearch/
chown -R elasticsearch:elasticsearch /etc/sysconfig/elasticsearch/
```

- Creare ora il servizio attraverso il quale poter monitorare Elastic Search:

```
vi /etc/systemd/system/elasticsearch.service
```

- Incollare in esso il seguente contenuto:

```
[Unit]
Description=Elasticsearch
Documentation=http://www.elastic.co
Wants=network-online.target
After=network-online.target

[Service]
Environment=ES_HOME=/usr/share/elasticsearch
Environment=CONF_DIR=/etc/elasticsearch
Environment=DATA_DIR=/var/lib/elasticsearch
Environment=LOG_DIR=/var/log/elasticsearch
Environment=PID_DIR=/var/run/elasticsearch
EnvironmentFile=-/etc/sysconfig/elasticsearch

WorkingDirectory=/usr/share/elasticsearch

User=elasticsearch
Group=elasticsearch

ExecStartPre=/usr/share/elasticsearch/bin/elasticsearch-systemd-pre-exec

ExecStart=/usr/share/elasticsearch/bin/elasticsearch \
                                                -p
${PID_DIR}/elasticsearch.pid \
                                                --quiet \
                                                -Edefault.path.logs=$
{LOG_DIR} \
                                                -Edefault.path.data=$
{DATA_DIR} \
                                                -Edefault.path.conf=$
{CONF_DIR}

# StandardOutput is configured to redirect to journalctl since
# some error messages may be logged in standard output before
# elasticsearch logging system is initialized. Elasticsearch
# stores its logs in /var/log/elasticsearch and does not use
# journalctl by default. If you also want to enable journalctl
# logging, you can simply remove the "quiet" option from ExecStart.
StandardOutput=journal
StandardError=inherit

# Specifies the maximum file descriptor number that can be opened by this
process
LimitNOFILE=65536

# Specifies the maximum number of bytes of memory that may be locked into RAM
```

```
# Set to "infinity" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml and 'MAX_LOCKED_MEMORY=unlimited' in
/etc/sysconfig/elasticsearch
LimitMEMLOCK=infinity

# Disable timeout logic and wait until process is stopped
TimeoutStopSec=0

# SIGTERM signal is used to stop the Java process
KillSignal=SIGTERM

# Java process is never killed
SendSIGKILL=no

# When a JVM receives a SIGTERM signal it exits with code 143
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target

# Built for distribution-5.4.1 (distribution)
```

- Dare maggiore disponibilità di memoria al servizio:

```
sysctl -w vm.max_map_count=262144
```

- Avviare ora il servizio:

```
systemctl start elasticsearch
```

3.5 Manuale Installazione Componente BigData

Le macchine iter-iterml1/2/3 prevedono:

- L'installazione della versione 2.7.3 del prodotto Hadoop data-node.
- L'installazione della versione 2.3.3 del prodotto Spark slave.
- L'installazione della versione 3.4.6 del prodotto Zookeeper.
- L'installazione della versione 0.208 del prodotto Presto worker.

La macchina iter-iterml4 prevede:

- L'installazione della versione 2.7.3 del prodotto Hadoop name-node.
- L'installazione della versione 2.3.3 del prodotto Spark master.

La macchina iter-iterml5

- L'installazione della versione 2.7.3 del prodotto Hadoop name-node.
- L'installazione della versione 2.3.3 del prodotto Spark master.
- L'installazione della versione 1.4.7 del prodotto Sqoop.
- L'installazione della versione 2.3.5 del prodotto Hive.
- L'installazione della versione 0.208 del prodotto Presto coordinator.
- Installazione di Supervisor

Al momento il bilanciamento tra i due name node è disabilitato. Nell'Ha-proxy dell'iter-is1 il puntamento a iter-ml4 è commentato in quanto, se la richiesta passa per il name node4, zeppelin potrebbe dare un eccezione.

Eccezione:

```
WARN [2019-09-09 10:48:20,827] ({pool-2-thread-6}
NotebookServer.java[afterStatusChange]:2302) - Job 20190904-143616_1459898634
is
finished, status: ERROR, exception: null, result: %text Py4JJavaError: An
error occurred while calling o874.showString.
:
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.ipc.StandbyException)
: Operation category READ is not supported in state s
tandby
        at
org.apache.hadoop.hdfs.server.namenode.ha.StandbyState.checkOperation(Standby
State.java:87)
        at
org.apache.hadoop.hdfs.server.namenode.NameNode$NameNodeHAContext.checkOperat
ion(NameNode.java:1779)
        at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkOperation(FSNamesyst
em.java:1313)
        at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getFileInfo(FSNamesystem.
java:3852)
        at
org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.getFileInfo(NameNode
RpcServer.java:1012)
        at
org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolServerSideTranslatorP
B.getFileInfo(ClientNamenodeProtocolServerSi
deTranslatorPB.java:843)
```

3.5.1 Installazione Zeppelin

Zeppelin è stato installato sulla macchina iter-iterml1 seguendo i passaggi elencati.

- Scaricare la versione di Zeppelin dal sito ufficiale di apache con il comando seguente:

```
wget http://apache.panu.it/zeppelin/zeppelin-0.8.0/zeppelin-0.8.0-bin-all.tgz
```

- Scompattare il file tramite il comando:

```
tar xfvz zeppelin-0.8.0-bin-all.tgz
```

- Creare la cartella al percorso seguente:

```
mkdir /opt/zeppelin/
```

- Spostare la cartella di Zeppelin sotto questo percorso

```
mv zeppelin-0.8.0-bin-all /opt/zeppelin/
```

- Creare un link simbolico che punti ad essa:

```
ln -s /opt/zeppelin-0.8.0-bin-all/ current
```

- Creare il servizio per zeppelin:

```
vi /etc/systemd/system/zeppelin.service
```

1. Ed incollare nell'editor le seguenti righe:

```
[Unit]
Description=Service to run Zeppelin Daemon
Documentation=

[Service]
User=analytics
Group=analytics
Type=forking
WorkingDirectory=/opt/zeppelin/current
ExecStart=/opt/zeppelin/current/bin/zeppelin-daemon.sh start
ExecStop=/opt/zeppelin/current/bin/zeppelin-daemon.sh stop

[Install]
WantedBy=multi-user.target
```

3.5.2 Installazione Anomaly Server

Per l'installazione di Supervisor ed Anomaly Server è stato utilizzato uno script Ansible, illustrato di seguito.

Al fine di una corretta implementazione dei prodotti modificare il seguente file per rendere le configurazioni congrue al sistema che si sta utilizzando.

Per lanciare lo script Ansible e quindi avviare l'installazione di anomaly Server occorre prima di tutto essere sulla stessa rete della macchina, quindi è necessario collegarsi alla VPN in questione.

Ci troveremo nella situazione in cui l'alberatura Ansible sarà sul nostro PC quindi dal teminale della

nostra macchina daremo:

```
cd /mnt/c/Users/NOME_UTENTE/Desktop/NOME_CARTELLA_ANSIBLE
```

Una volta dentro la cartella, daremo il comando per avviare l'installazione:

```
ansible-playbook -i inventories/iter-prod/hosts setup.yml
```

Lo script partirà eseguendo uno dopo l'altro i vari task richiamati dal Role e dall'Hosts.

3.5.3 Variabili Ansible

Nel file /iter-prod/hosts oltre ad avere gli indirizzi delle VM sulle quali andremo ad installare il componente, troveremo alcuni parametri fondamentali da modificare rispetto all'ambiente in cui siamo.

Nel file host, le variabili saranno semplicemente identificabili in quanto sono precedute dal blocco [all:vars].

Modificare la variabile nel seguente file:

inventories/iter-prod/hosts

```
ansible_user=centos
```

Modificare le variabili nel seguente file:

roles/anomaly-server/defaults/main.yml

```
#user
anomaly_user: root

#folders
anomaly_server_installation_path: /opt/anomaly-server
anomaly_server_config_path: /etc/anomaly-server

git_branch: master
git_repo: git@gitlab.com:giotto-machine-learning/anomaly-server.git
```

```
spark_master: local
hadoop_name_node: localhost:8020
spark_installation_path: /opt/spark
spark_config_path: /etc/spark

_common_input_dataset_path: "hdfs://{{ hadoop_name_node }}/spark/datasets"
_common_input_model_path: "hdfs://{{ hadoop_name_node }}/spark/models"

layoff_input_dataset_path: "{{ _common_input_dataset_path }}/layoff.csv"
layoff_input_model_path: "{{ _common_input_model_path }}/layoff_model"

journey_input_dataset_path: "{{ _common_input_dataset_path }}/journey.csv"
journey_input_model_path: "{{ _common_input_model_path }}/journey_model"

anomaly_server_log_path: /var/log/anomaly-server

# supervisord variables
supervisord_log_dir: /var/log/supervisor
supervisord_run_dir: /var/run/supervisor
supervisord_port: 9001

#broker confg
broker_jms_host: localhost
broker_jms_port: 61613

broker_jms_user: admin
broker_jms_password: passwordscelta

broker_jms_input_queue: /queue//queue/positions

broker_mqtt_host: localhost
broker_mqtt_port: 1883

broker_mqtt_user: admin
broker_mqtt_password: passwordscelta

broker_mqtt_output_topic: GEO/ANOMALIES

deploy_from_git: yes
```

3.5.4 Monitoraggio

3.5.5 Avvio Servizi

NOME VM	RUOLO	IP	HOST ANSIBLE	COMPONENTI	SYSTEMD
Iter-iternml5	namenode 1 (*primario*)	10.244.10.33	*hive *sqoop *master-	*(HADOOP/POSTGRESQL-CLIENT, HIVE-METASTORE, HADOOP/HIVE 2.3.5, INSTALL-PRESTO-COORDINATOR 0.208), *SQOOP 1.4.7, *(HADOOP/NAME-NODE 2.7.3, SPARK/CLUSTER-MASTER 2.3.3)	sudo systemctl start hadoop-name-node.service sudo systemctl start hadoop-zkfc.service sudo systemctl start yarn-nodemanager.service

			nodes		sudo systemctl start resource sudo systemctl start spark-master
Iter-iternml4	namenode 2	10.244.10.32	*master-nodes	*(HADOOP/NAME-NODE 2.7.3, SPARK/CLUSTER-MASTER 2.3.3)	***Vedi namenode1***
Iter-iternml1	datanode1	10.244.10.29	*data-nodes *zookeeper	*(HADOOP/DATA-NODE 2.7.3, SPARK/STANDALONE-SLAVE 2.3.3, PRESTO-WORKER 0.208) *ZOOKEEPER 3.4.6	sudo systemctl start zookeeper sudo systemctl start hadoop-journal-node.service sudo systemctl start hadoop-data-node.service sudo systemctl start spark-slave sudo systemctl start yarn-nodemanager.service
Iter-iternml2	datanode2	10.244.10.30	*data-nodes *zookeeper	*(HADOOP/DATA-NODE 2.7.3, SPARK/STANDALONE-SLAVE 2.3.3, PRESTO-WORKER 0.208) *ZOOKEEPER 3.4.6	***Vedi datanode1***
Iter-iternml3	datanode3	10.244.10.31	*data-nodes *zookeeper	*(HADOOP/DATA-NODE 2.7.3, SPARK/STANDALONE-SLAVE 2.3.3, PRESTO-WORKER 0.208) *ZOOKEEPER 3.4.6	***Vedi datanode1***

3.5.6 Sequenza di avvio dei servizi

Per il corretto avvio dell'ambiente, eseguire gli script di startup nel seguente ordine:

Data Node

```
sudo systemctl start zookeeper
sudo systemctl start hadoop-journal-node.service
sudo systemctl start hadoop-data-node.service
```

Name Node

```
sudo systemctl start hadoop-name-node.service
sudo systemctl start hadoop-zkfc.service
sudo systemctl start resource
```

Name Node e Data Node

```
sudo systemctl start yarn-nodemanager.service
```

Nel caso in cui i servizi **Hive** non fossero già avviati eseguire gli script sul **Name Node1**:

```
systemctl start hive-server.service
systemctl start hive-metastore.service
```

Nel caso in cui i servizi **Spark** non fossero già avviati eseguire gli script:

```
systemctl start spark-master      sui 2 Name Node
systemctl start spark-slave       sui 3 Data Node
```

3.5.7 JPS

Il comando da utilizzare per monitorare i servizi attivi dell'ecosistema big data è:

- /usr/java/jdk1.8.0_181-amd64/bin/jps

L'output per ogni macchina deve essere:

- iter-ml1

```
5104 DataNode
977 ZeppelinServer
5522 Worker
5622 Jps
4839 QuorumPeerMain
5335 NodeManager
3018 SparkSubmit
699 PrestoServer
5003 JournalNode
```

- Iter-ml2

```
3414 Jps
4600 JournalNode
5112 Worker
697 PrestoServer
4697 DataNode
4923 NodeManager
4511 QuorumPeerMain
```

- Iter-ml3

```
705 PrestoServer
15398 JournalNode
15718 NodeManager
15495 DataNode
14344 Jps
15981 CoarseGrainedExecutorBackend
15310 QuorumPeerMain
15903 Worker
```

- Iter-ml4

```
8192 NodeManager
7973 NameNode
```

```
8087 DFSZKFailoverController
8348 Master
26092 Jps
```

- Iter-ml5

```
24577 PrestoServer
11570 RunJar
11699 RunJar
10851 NameNode
11587 NodeManager
11303 ResourceManager
11801 Master
24907 Jps
10942 DFSZKFailoverController
```

3.5.8 Comandi utilli

- Start Resource Manager

```
systemctl start yarn-resource
```

- Stop Resource Manager

```
systemctl stop  yarn-resource
```

- hdfs dfsadmin -report

```
hdfs dfsadmin -report
Configured Capacity: 1181081341952 (1.07 TB)
Present Capacity: 1161200029696 (1.06 TB)
DFS Remaining: 1132683841536 (1.03 TB)
DFS Used: 28516188160 (26.56 GB)
DFS Used%: 2.46%
Under replicated blocks: 0
Blocks with corrupt replicas: 0
Missing blocks: 0
Missing blocks (with replication factor 1): 0

-------------------------------------------------
Live datanodes (3):

Name: 10.244.10.29:50010 (iter-ml1)
Hostname: iter-iterml1.novalocal
Decommission Status : Normal
Configured Capacity: 322110992384 (299.99 GB)
DFS Used: 9505394688 (8.85 GB)
```

```
Non DFS Used: 12111011840 (11.28 GB)
DFS Remaining: 300494585856 (279.86 GB)
DFS Used%: 2.95%
DFS Remaining%: 93.29%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Fri Sep 06 13:12:07 UTC 2019


Name: 10.244.10.30:50010 (iter-ml2)
Hostname: iter-iterml2.novalocal
Decommission Status : Normal
Configured Capacity: 322110992384 (299.99 GB)
DFS Used: 9505398784 (8.85 GB)
Non DFS Used: 3893506048 (3.63 GB)
DFS Remaining: 308712087552 (287.51 GB)
DFS Used%: 2.95%
DFS Remaining%: 95.84%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Fri Sep 06 13:12:08 UTC 2019


Name: 10.244.10.31:50010 (iter-ml3)
Hostname: iter-iterml3.novalocal
Decommission Status : Normal
Configured Capacity: 536859357184 (499.99 GB)
DFS Used: 9505394688 (8.85 GB)
Non DFS Used: 3876794368 (3.61 GB)
DFS Remaining: 523477168128 (487.53 GB)
DFS Used%: 1.77%
DFS Remaining%: 97.51%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Fri Sep 06 13:12:07 UTC 2019
```

3.6 INSTALLAZIONE WSO2ESB

Le macchine iter-esb1 e iter-esb2 prevedono l'installazione dei seguenti prodotti:

- JDK 1.8
- Wso2esb 5.0.0

3.6.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

3.6.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

3.6.3 Installazione wso2esb

Per l'installazione del software wso2esb è stato utilizzato il pacchetto, fornito, di nome **iteriot-esb.tar.gz**.

Eseguire un trasferimento tramite protocollo SCP e una volta che il pacchetto è stato scaricato sulla macchina procedere come segue.

- Creare una cartella sotto il seguente path:

```
mkdir /opt/wso2esb-5.0.0/
```

- Muovere il file compresso nella cartella appena creata:

```
mv iteriot-esb.tar.gz /opt/wso2esb-5.0.0
```

- Scompattare il file:

```
tar xfvz iteriot-esb.tar.gz
```

- Creare il servizio per il software wso2esb al seguente path:

```
vi /etc/systemd/system/wso2.service
```

- Incollare in questo file quanto segue:

```
[Unit]
Description=WSO2 Enterprise Service Bus
After=syslog.target network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre
ExecStart=/opt/wso2esb-5.0.0/bin/wso2server.sh start
ExecStop=/opt/wso2esb-5.0.0/bin/wso2server.sh stop
#RemainAfterExit=yes
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target
Avviare il servizio:
systemctl start wso2esb
```

3.6.4 Clusterizzazione nodi ESB

Su entrambi i nodi eseguire:

- Installare corosync

```
yum install corosync pcs pacemaker
```

- Abilitare l'avvio allo start delle macchine

```
systemctl enable pcsd
systemctl start pcsd
```

- Cambiare la password per l'utente "hacluster"

```
passwd hacluster
```

- Come output avremo le seguenti righe:

```
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

- Autorizzare i nodi a partecipare al cluster:

```
pcs cluster auth iter-esb1.regione.campania.it iter-esb2.regione.campania.it
```

- Risultato:

```
Username: hacluster
Password:
iter-esb1.regione.campania.it: Authorized
iter-esb2.regione.campania.it: Authorized
```

- Creazione del cluster di nome “ha_cluster”:

```
pcs cluster setup --name ha_cluster iter-esb1.regione.campania.it iter-
esb2.regione.campania.it
```

- Avviare il tutto:

```
pcs cluster start --all
```

- Settare i parametri del cluster:

```
pcs property set stonith-enabled=false
pcs property set no-quorum-policy=ignore
pcs property set default-resource-stickiness="INFINITY"
```

- Controllare il cambio delle modifiche apportate precedentemente

```
pcs property list
```

- Risultato:

```
Cluster Properties:
cluster-infrastructure: corosync
cluster-name: ha_cluster
dc-version: 1.1.19-8.el7_6.4-c3c624ea3d
default-resource-stickiness: INFINITY
have-watchdog: false
no-quorum-policy: ignore
stonith-enabled: false
```

- Concludere clusterizzando entrambi i nodi:

```
pcs resource create ESB_service systemd:wso2esb op monitor interval="10s"
timeout="15s" op start interval="0" timeout="15s" op stop interval="0"
timeout="30s"
```

3.7 INSTALLAZIONE WSO2CEP

Le macchine iter-cep1 e iter-cep2 prevedono l’installazione dei seguenti prodotti:

- JDK 1.8
- Wso2cep 4.2.0

3.7.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati

- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

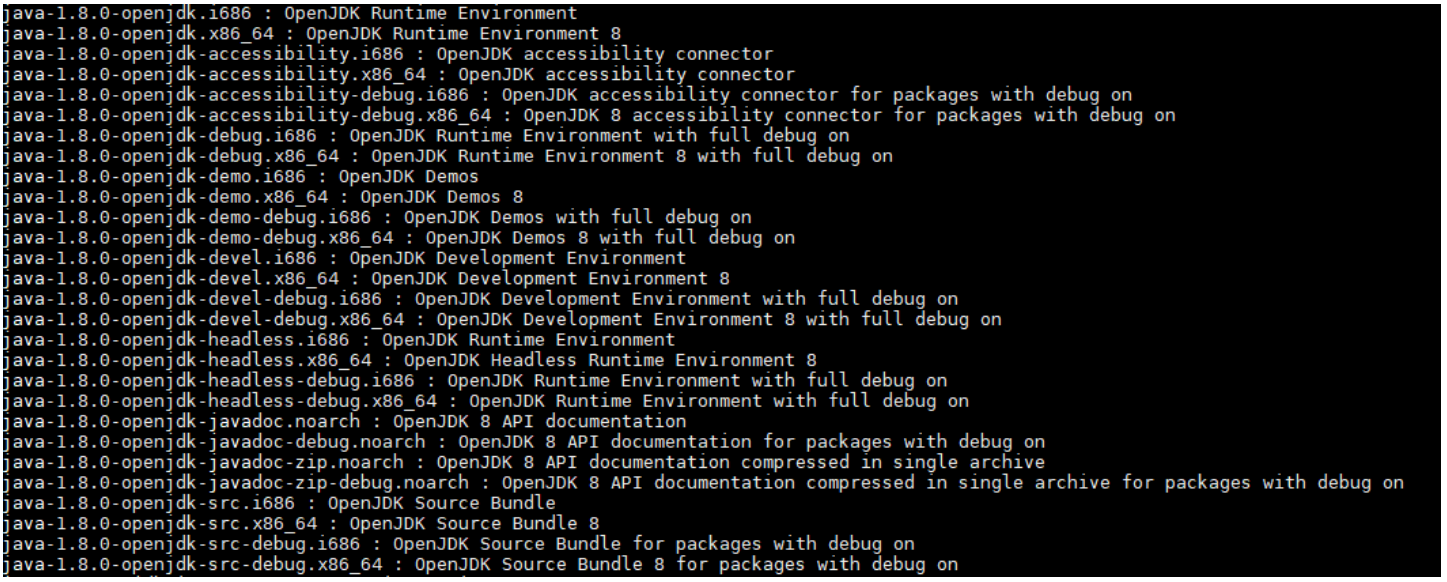
3.7.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:



- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

3.7.3 Installazione wso2cep

Per l'installazione del software wso2esb è stato utilizzato il pacchetto, fornito, di nome **wso2cep-4.2.0-bak.tar.gz**. Eseguire un trasferimento tramite protocollo SCP e una volta che il pacchetto è stato scaricato sulla macchina procedere come segue.

- Creare una cartella sotto il seguente path:

```
mkdir /opt/wso2cep-4.2.0/
```

- Muovere il file compresso nella cartella appena creata:

```
mv wso2cep-4.2.0-bak.tar.gz /opt/wso2cep-4.2.0
```

- Scompattare il file:

```
tar xfvz wso2cep-4.2.0-bak.tar.gz
```

- Creare il servizio per il software wso2cep al seguente path:

```
vi /etc/systemd/system/wso2cep.service
```

- Incollare in questo file quanto segue:

```
[Unit]
Description=WSO2 Complex Event Processor
After=syslog.target network.target

[Service]
Type=forking
```

```
Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre
ExecStart=/opt/wso2cep-4.2.0/bin/wso2server.sh start
ExecStop=/opt/wso2cep-4.2.0/bin/wso2server.sh stop
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target
```

3.7.4 Clusterizzazione Nodi CEP

Su entrambi i nodi eseguire:

- Installiamo corosync

```
yum install corosync pcs pacemaker
```

- Abilitiamo l'avvio allo start delle macchine

```
systemctl enable pcsd
systemctl start pcsd
```

- Cambiamo la password per l'utente "hacluster"

```
passwd hacluster
```

- Risultato:

```
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

- Autorizziamo i nodi a partecipare al cluster:

```
pcs cluster auth iter-cep1.regione.campania.it iter-cep2.regione.campania.it
```

- Risultato:

```
Username: hacluster
Password:
iter-cep2.regione.campania.it: Authorized
iter-cep1.regione.campania.it: Authorized
```

- Creazione del cluster di nome "ha_cluster"

```
pcs cluster setup --name ha_cluster iter-cep1.regione.campania.it iter-
cep2.regione.campania.it
```

- Avviamo tutto

```
pcs cluster start --all
```

- Settiamo i parametri del cluster

```
pcs property set stonith-enabled=false
pcs property set no-quorum-policy=ignore
pcs property set default-resource-stickiness="INFINITY"
```

- Controlliamo il cambio delle modifiche apportate precedentemente

```
pcs property list
```

- Risultato:

```
Cluster Properties:
cluster-infrastructure: corosync
cluster-name: ha_cluster
dc-version: 1.1.19-8.el7_6.4-c3c624ea3d
default-resource-stickiness: INFINITY
have-watchdog: false
no-quorum-policy: ignore
stonith-enabled: false
```

- Concludiamo clusterizzando entrambi i nodi:

```
pcs resource create CEP_service systemd:wso2cep op monitor interval="10s"
timeout="15s" op start interval="0" timeout="15s" op stop interval="0"
timeout="30s"
```

3.8 INSTALLAZIONE WSO2IS

Le macchine iter-is1 e iter-is2 prevedono l'installazione dei seguenti prodotti:

- JDK 1.8
- Haproxy 1.5.18
- Apache 2.4.6
- MariaDB 5.5

3.8.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

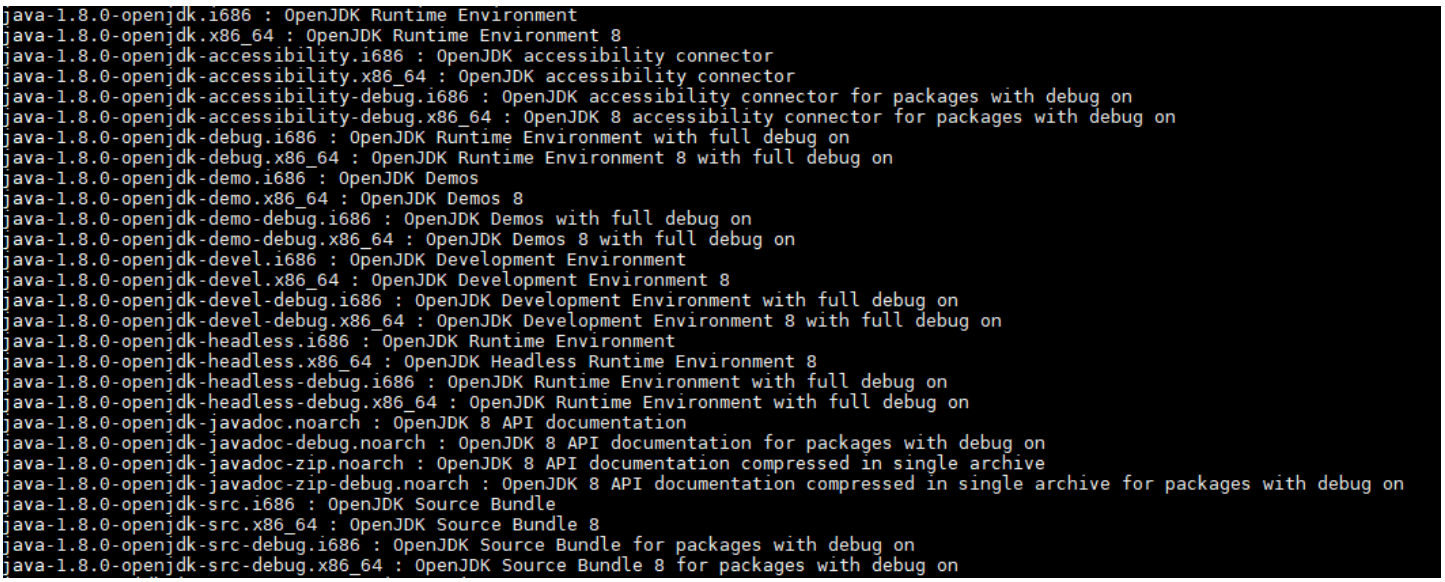
3.8.2 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:



- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.el7_6.x86_64/jre/
```

3.8.3 Installazione haproxy

Per l'installazione di haproxy procedere come di seguito.

- Installare il pacchetto:

```
yum install haproxy
```

- Modificare il contenuto della configurazione di haproxy nel file:

```
vi /etc/haproxy/haproxy.cfg
```

- Incollare all'interno il seguente contenuto:

```
global
    log         127.0.0.1 local2

    chroot      /var/lib/haproxy
    pidfile     /var/run/haproxy.pid
    maxconn     4000
    user        haproxy
    group       haproxy
    daemon

    # turn on stats unix socket
    stats socket /var/lib/haproxy/stats

defaults
    log                     global
    option                  dontlognull
    option http-server-close
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          1m
    timeout server          1m
    timeout http-keep-alive 10s
    timeout check           10s
    maxconn                 3000
listen  stats :9000
        stats enable
        stats realm Haproxy\ Statistics
```

```
        stats uri /haproxy_stats
        stats auth admin:password
        stats refresh 30
        mode http

frontend  elastic *:9200
    default_backend             mgmt9200

backend mgmt9200
    balance roundrobin
    mode http
    option tcpka
        server iter-data1.regione.campania.it iter-
data1.regione.campania.it:9200 check
        server iter-data2.regione.campania.it iter-
data2.regione.campania.it:9200 check
        server iter-data3.regione.campania.it iter-
data3.regione.campania.it:9200 check

frontend  activemq-mqtt *:1883
    default_backend             mgmt1883
    timeout client 3h
    option tcplog
    option clitcpka

backend mgmt1883
    balance source
    mode tcp
    option srvtcpka
    option redispatch
    timeout server 3h
    timeout connect 5000ms
        server iter-mqtt1.regione.campania.it iter-
mqtt1.regione.campania.it:1883 check
        server iter-mqtt2.regione.campania.it iter-
mqtt2.regione.campania.it:1883 check backup
        server iter-mqtt3.regione.campania.it iter-
mqtt3.regione.campania.it:1883 check backup

frontend  activemq-openwire *:61616
    default_backend             mgmt61616
    timeout client 3h
    option tcplog
    option clitcpka

backend mgmt61616
    balance source
    mode tcp
    option srvtcpka
    timeout server 3h
    timeout connect 5000ms
        server iter-mqtt1.regione.campania.it iter-
mqtt1.regione.campania.it:61616 check
        server iter-mqtt2.regione.campania.it iter-
```

```
mqtt2.regione.campania.it:61616 check backup
        server iter-mqtt3.regione.campania.it iter-
mqtt3.regione.campania.it:61616 check backup

frontend  activemq-stomp *:61613
    default_backend             mgmt61613
    timeout client 3h
    option tcplog
    option clitcpka

backend mgmt61613
    balance source
    mode tcp
    option srvtcpka
    timeout server 3h
    timeout connect 5000ms
        server iter-mqtt1.regione.campania.it iter-
mqtt1.regione.campania.it:61613 check
        server iter-mqtt2.regione.campania.it iter-
mqtt2.regione.campania.it:61613 check backup
        server iter-mqtt3.regione.campania.it iter-
mqtt3.regione.campania.it:61613 check backup

frontend  fleetmgmt *:82
    default_backend             backend_fleetmgmt

backend backend_fleetmgmt
    balance roundrobin
    mode http
    option tcpka
        server iterfleet iterfleetservice.regione.campania.it:8080 check

frontend  fleetwapp *:81
    default_backend             backend_fleetwapp

backend backend_fleetwapp
    balance roundrobin
    mode http
    option tcpka
        server iter-gfleet-webapp iterfleetweb.regione.campania.it:8080 check
```

- Avviare il servizio affinché le regole di load balancing siano attive:

```
systemctl start haproxy
```

3.8.4 Installazione httpd

Per l'installazione del software procedere come descritto di seguito.

- Scaricare il pacchetto con il comando seguente:

```
yum install httpd
```

Per la configurazione di httpd sono stati usati gli stessi file utilizzati in CRED. Per cui seguire i passi

elencati.

- Rimuovere ora le cartelle installate automaticamente dal gestore yum:

```
rm -rf /etc/httpd/conf.d/ /var/www/html
```

- Copiare tramite protocollo SCP le stesse cartelle da CRED a CLOUD le quali hanno già le configurazioni all'interno:

```
/etc/httpd/conf.d/
/var/www/html/
```

- Modificare tutti i file contenuti nella directory conf.d. Questi file contengono i puntamenti alle altre VM dell'ambiente. Sostituire perciò tutti gli indirizzi o i nomi host con i rispettivi che si trovano in CLOUD.

1. Installare il pacchetto modulo ssl per il corretto funzionamento del software:

```
yum install mod_ssl
```

- Riavviare adesso il servizio rendendo effettive le modifiche:

```
systemctl restart httpd
```

3.9 INSTALLAZIONE SENSORI

Le macchine iter-sensor1 e iter sensori3 prevedono l'installazione dei seguenti prodotti:

- JDK 1.8
- Nifi 1.9.2

La macchina iter-sensori2 prevede l'installazione dei seguenti prodotti:

- JDK 1.8
- Nifi 1.9.2
- Redash 4.0

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

3.9.1 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-
```

```
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.el7_6.x86_64/jre/
```

3.9.2 Installazione Nifi

Per l'installazione del software è necessario scaricare la versione desiderata dal sito ufficiale del prodotto.

- Installare wget:

```
yum install wget
```

- Scaricare Nifi 1.9 con il seguente comando nella home dell'utente:

```
cd /home/centos
wget http://www.apache.org/dist/nifi/1.9.2/nifi-1.9.2-bin.tar.gz
```

- Scompattare il file e spostarlo sotto il percorso /opt/

```
tar xfvz nifi-1.9.2-bin.tar.gz
mv nifi-1.9.2 /opt/
mv nifi-1.9.2/ nifi/
```

- Creazione del servizio per il monitoraggio di nifi:

```
vi /etc/systemd/system/nifi.service
```

- Incollare in quest'ultimo le seguenti righe:

```
[Unit]
Description=Apache NiFi
After=network.target

[Service]
Type=forking
User=root
Group=root
ExecStart=/opt/nifi/bin/nifi.sh start
ExecStop=/opt/nifi/bin/nifi.sh stop
#ExecRestart=/opt/nifi/bin/nifi.sh restart

[Install]
WantedBy=multi-user.target
```

- Creazione della cartella che ospiterà le configurazioni di zookeeper:

```
mkdir /opt/nifi/state
mkdir /opt/nifi/state/zookeeper
```

- Creare un file con il numero identificativo del nodo per zookeeper:

```
echo 1 > /opt/nifi/state/zookeeper/myid
```

- Modifica del file nifi.properties:

```
vi /opt/nifi/conf/nifi.properties
```

- Cercare nel suddetto file le seguenti voci e modificarle come indicato di seguito:

```
nifi.state.management.embedded.zookeeper.start=true
nifi.web.http.host=iter-sensori1
nifi.cluster.is.node=true
nifi.cluster.node.address=iter-sensori1
nifi.cluster.node.protocol.port=11433
nifi.zookeeper.connect.string=iter-sensori1:2181,iter-sensori2:2181,iter-
sensori3:2181
```

- Modifica del file zookeeper.properties:

```
vi /opt/nifi/conf/zookeeper.properties
```

- Aggiungere nel suddetto file le seguenti voci per l'identificazione dei server:

```
server.1=iter-sensori1:2888:3888
server.2=iter-sensori2:2888:3888
server.3=iter-sensori3:2888:3888
```

- Modifica del file state-management.xml:

```
vi /opt/nifi/conf/state-management.xml
```

- Cercare la voce seguente e modificarla come segue:

```
<property name="Connect String">iter-sensori1:2181,iter-sensori2:2181,iter-
sensori3:2181</property>
```

- Per lo start del servizio utilizzare il seguente comando:

NB: avviare il servizio su tutte le macchine contemporaneamente.

```
systemctl start nifi
```

- Al fine di controllare che l'avvio stia andando a buon fine è possibile guardare il file di log con il seguente comando:

```
tail -f /opt/nifi/logs/nifi-app.log
```

Una volta avviati i tre nodi, come si può vedere dai logs, si avvierà il processo di zookeeper per l'elezione del coordinatore. Finita l'elezione sarà possibile accedere all'interfaccia web del prodotto semplicemente collegandosi al seguente indirizzo:

```
http://10.244.10.21:8080/nifi
```

3.9.3 Comandi per la gestione del servizio

- Avvio del servio:

```
systemctl start nifi
```

- Stop del servizio:

```
systemctl stop nifi
```

- Check del servizio:

```
systemctl status nifi
```

- Restart del servizio

```
systemctl restart nifi
```

3.9.4 Installazione Redash

Per l'installazione di Redash è stato utilizzato uno script Ansible per automatizzare il processo.

- Per lanciare lo script Ansible occorrerà installare sulla macchina in cui si trova lo script, Ansible e Python almeno 2.7 come pre-requisito. Troveremo il tutto al seguente link che avrà a disposizione l'RPM di installazione di Ansible 2.8.0 per i diversi sistemi operativi e Python:

```
https://pkgs.org/download/ansible
```

Una volta effettuata l'installazione, bisognerà modificare il file inventory/host.ini e cambiare gli indirizzi di IP sui quali far girare lo script ansible. Inoltre, prima di avviare l'ansible, dovremo accedere via SSH ad ogni macchina su cui avverrà l'installazione.

- Creazione delle chiavi pubblica/privata sulla macchina sorgente

(dove abbiamo installato ansible e python)

- Accedere alla macchina come root e dare il comando:

```
ssh-keygen
```

Dare invio per i settaggi di default. Avrà creato sotto .ssh

id_rsa (chiave privata)

id_rsa.pub (chiave pubblica)

Creazione delle chiavi pubblica/privata sulle tutte le macchine di destinazione.

(dove dovrà girare l'ansible)

- Accedere alla macchina destinataria come root e generare le chiavi pubblica e privata e dare:

```
ssh-keygen
```

- Dare invio per i settaggi di default. Accedere poi alla directory .ssh con cd .ssh e creare il file authorized_keys nel quale andremo ad incollare la publicKey precedentemente creata dal nostro pc *(dove abbiamo installato ansible e python)*

```
vi authorized_keys
```

e incollare la chiave pubblica.

- Diamo infine i permessi a .ssh e al file authorized_keys:

```
chmod 700 .ssh
chmod 0600 authorized_keys
```

- Sempre sulla macchina sulla quale avremo installato ansible e python effettueremo gli accessi ssh verso le macchine di destinazione:
  Esempio:

```
ssh Nome_Utente@IP_macchina_destinazione
```

L'alberatura dello script ansible si suddivide in questo modo:

Esempio:

Nella cartella **inventories** troviamo il file iter.ini che ci mostra su quali macchine verranno installati i componenti.

Esempio:

```
[redash]
redash ansible_ssh_host=10.244.10.22 private_ip=10.244.10.22
```

Nel file **setup.yml** vengono definiti i componenti (chiamati **role**) da installare per ogni Hosts.

Nella cartella **roles**, troviamo quindi tutti i componenti che verranno installati lanciando l'ansible.

A loro volta, all'interno troveremo una directory chiamata **tasks** che conterrà un file **.yml** con tutti i passaggi che eseguirà lo script affinché si possa installare automaticamente il componente.

3.9.5 Avvio Script Ansible

- Per lanciare lo script Ansible e quindi avviare l'installazione dei componenti sulle varie macchine occorre prima di tutto essere sulla stessa rete delle macchine, quindi è necessario collegarsi alla VPN in questione. Ci troveremo nella situazione in cui l'alberatura Ansible sarà sul nostro PC quindi dal teminale della nostra macchina daremo:

```
cd /mnt/c/Users/NOME_UTENTE/Desktop/NOME_CARTELLA_ANSIBLE
```

- Una volta dentro la cartella, daremo:

```
ansible-playbook -i inventories/iter.ini setup.yml
```

Lo script partirà eseguendo uno dopo l'altro i vari task richiamati dal Role e dall'Hosts.

3.9.6 Correzioni

Lo script Ansible potrebbe dare problemi di dipendenze.

Le correzioni da fare in corso d'opera sono le seguenti.

- Nel momento in cui l'Ansible darà errore per ciò che riguarda la libreria wsogiref e/o memsql, basterà modificare il file seguente:

```
vi /opt/redash/requirements_all_ds.txt
```

- Eliminare le suddette librerie dal file.
- Nel momento in cui l'Ansible termina e Redash non riesce ad avviarsi controllare tramite il seguente comando la versione della libreria installata:

```
pip freeze | grep oauthlib
```

- La versione corretta è la 2.1.0. Se la versione è diversa usare il seguente comando:

```
pip install oauthlib==2.1.0
```

- Restartare il servizio con il comando:

```
systemctl restart supervisord
```

- Per controllare i file di log in tempo reale utilizzare:

```
tail -f /var/log/supervisor/supervisord.log
```

Per il corretto funzionamento di Redash è necessario che il database PostgreSQL abbia la codifica UTF-8.

Se c'è bisogno di modificare tale codifica eseguire i seguenti punti.

- Fare login come root:

```
sudo su
```

- Fare login come postgres:

```
su - postgres
```

- Entrare nel database con il seguente comando:

```
psql
```

- Eseguire il comando:

```
update pg_database set encoding = pg_char_to_encoding('UTF8') where datname =
'redash';
```

3.9.7 Monitoraggio

Per l'accesso alla GUI le credenziali di accesso sono:

- email: superuser@superuser.it
- password: superuser

3.10 Installazione NFS Server

La macchina iter-nfs prevede l'installazione dei seguenti prodotti:

- NFS 1.3.0

3.10.1 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

3.10.2 Installazione NFS

Per l'installazione del prodotto utilizzare il gestore yum come di seguito:

```
yum install nfs-utils
```

Estensione del disco

```
fdisk -l
lvextend -l +100%FREE /dev/VGdata/LVdata
pvresize /dev/vdb
xfs_growfs /dev/VGdata/LVdata -d
cd /export-fs/data/
mkdir SHARE
cd SHARE
mkdir ESB
mkdir CEP
chmod 777 ESB/ CEP/ -R
vi /etc/exports
/export-fs/data/SHARE/CEP
10.244.10.17/32(rw,no_root_squash,async,no_subtree_check)
/export-fs/data/SHARE/CEP
10.244.10.18/32(rw,no_root_squash,async,no_subtree_check)
/export-fs/data/SHARE/ESB
10.244.10.19/32(rw,no_root_squash,async,no_subtree_check)
```

```
/export-fs/data/SHARE/ESB
10.244.10.20/32(rw,no_root_squash,async,no_subtree_check)
/export-fs/data/SHARE/IS
10.244.10.27/32(rw,no_root_squash,async,no_subtree_check)
/export-fs/data/SHARE/IS
10.244.10.28/32(rw,no_root_squash,async,no_subtree_check)
```

3.10.3 NFS Mount Point (CRED)

Nella seguente tabella sono riportate tutte le regole di import/export NFS. In verde le macchine che esportano le directory condivise.

Le altre VM descritte importano il servizio dalle macchine che lo espongono.

Export VM	Import VM /etc/exports	Dire ctory esportate	Mount Point	Opzioni	Memoria Utilizzata
Iter-hub1	10.200.19.0/24	/home/iterusr/ gs_data_dir_local	/home/iterusr/gs_data_dir_local	(rw,sync,no _root_squas h,no_all_sq uash)	1,3G
iter-is	Iter-cep1 Iter-cep2	/SHARE/CEP	/opt	(rw,sync,no _root_squas h,no_all_sq uash)	11GB
iter-is	iter-esb1 iter-esb2	/SHARE/ESB	/opt	(rw,sync,no _root_squas h,no_all_sq uash)	11GB
Iter-gis1	Iter-geo1	/home/iterusr/ tomcatwg/temp	/home/iterusr/tomcatwg/temp	(rw,sync,no _root_squas h,no_all_sq uash)	29 MB
gitsrv.camp ania.local	iter-form	/FORM_DATA	/home/moodledata		
gitsrv.camp ania.local	Iter-geo2	/geoportal	/home/iterusr/Geoportal		
gitsrv.camp ania.local	iter-com1	/geocache	/home/iterusr/geocache		
gitsrv.camp ania.local	Iter-gis1	/geoportal	/home/iterusr/Geoportal		
gitsrv.camp ania.local	Iter-gis2	/geoportal	/run/user/1000		
gitsrv.camp ania.local	Iter-hub1	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.camp	Iter-hub2	/geocache	/home/iterusr/geocache		

Export VM	Import VM /etc/exports	Dire ctory esportate	Mount Point	Opzioni	Memoria Utilizzata
ania.local		/geoportal /gs_data_dir /data	/home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub3	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub4	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub5	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub6	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub7	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		
gitsrv.campania.local	Iter-hub8	/geocache /geoportal /gs_data_dir /data	/home/iterusr/geocache /home/iterusr/Geoportal /home/iterusr/gs_data_dir /home/iterusr/data		

3.10.4 Bilanciatori

Le macchine saranno bilanciate seguendo questo schema e tramite l'indirizzo del balancer associato

Nome balancer	Vm bilanciate	Indirizzo IP	IP Balancer TIM	Porte da bilanciare
Iter-esb	Iter-esb1	10.244.10.19	10.244.10.42	41709 - 9999 - 8243 - 8280 - 9443 - 9763 - 11111 /tcp
	Iter-esb2	10.244.10.20	10.244.10.42	41709 - 9999 - 8243 - 8280 - 9443 - 9763 - 11111 /tcp
Iter-cep	Iter-cep1	10.244.10.17	10.244.10.44	9611 - 9999 - 9711 - 40849 - 7611 - 7711 - 9443 - 9763 - 11111 /tcp
	Iter-cep2	10.244.10.18	10.244.10.42	9611 - 9999 - 9711 - 40849 - 7611 - 7711 - 9443 - 9763 - 11111 /tcp
Iter-is	Iter-is1	10.244.10.27	10.244.10.46	9443 - 9999 - 41939 - 10389 - 39479 - 10711 - 9763 - 11111/tcp
	Iter-is2	10.244.10.28	10.244.10.46	9443 - 9999 - 41939 - 10389 - 39479 - 10711 - 9763 - 11111/tcp

Nome balancer	Vm bilanciate	Indirizzo IP	IP Balancer TIM	Porte da bilanciare
Iter-nifi:80	Iter-sensori1	10.244.10.21	10.244.10.48	8080/tcp
	Iter-sensori2	10.244.10.22	10.244.10.48	8080/tcp
	Iter-sensori3	10.244.10.23	10.244.10.48	8080/tcp

Nome balancer	Vm bilanciate	Indirizzo IP	IP Balancer TIM	Porte da bilanciare
Iter-elastic	Iter-data1	10.244.10.24	10.244.10.50	9200 - 9300/tcp
	Iter-data2	10.244.10.25	10.244.10.50	9200 - 9300/tcp
	Iter-data3	10.244.10.26	10.244.10.50	9200 - 9300/tcp

Nome balancer	Vm bilanciate	Indirizzo IP	IP Balancer TIM	Porte da bilanciare
Iter-mqtt	Iter-mqtt1	10.244.10.14	10.244.10.52	1883 - 8883 - 61616 - 8161 - 61613/tcp
	Iter-mqtt2	10.244.10.15	10.244.10.52	1883 - 8883 - 61616 - 8161 - 61613/tcp
	Iter-mqtt3	10.244.10.16	10.244.10.52	1883 - 8883 - 61616 - 8161 - 61613/tcp

3.11 INTERFACCE WEB

3.11.1 Machine Learning

- Resource Manager

```
http://10.244.10.33:8088
```

- Hadoop

```
http://10.244.10.33:50070
```

- Zeppelin

```
http://10.244.10.29:8090
```

- Spark

```
http://10.244.10.33:8080
```

- Anomaly Server

```
http://10.244.10.33:9001/
```

3.11.2 Elasticsearch

```
10.244.10.24:9200
```

3.11.3 WSO2CEP

```
https://10.244.10.17:9443
```

3.11.4 WSO2ESB

```
https://10.244.10.19:9443
```

3.11.5 WSO2IS

```
https://10.244.10.27:9443
```

3.11.6 GFLEET

```
https://10.244.10.37/gfleet
```

3.11.7 ActiveMQ

```
https://10.244.10.14:8161
```

3.11.8 Redash

```
http://10.244.10.22
```

3.11.9 Nifi

```
http://10.244.10.21:8080/nifi
```

4 ANALISI DEL CONTESTO NORMATIVO, REGOLAMENTARE, E ORGANIZZATIVO CONTENENTE LA MAPPATURA DEGLI AMBITI DI APPLICAZIONE DELLA BUONA PRATICA E IL WORK FLOW

Il Progetto, in linea con la Strategia Europea in materia di ITC declinata nell'OT 2, ha realizzato un modello E-Government puntando alla digitalizzazione dei processi amministrativi e la diffusione di servizi digitali a cittadini, imprese e Enti Territoriali in coerenza con il R.A. 2.2. dell'OT2. Il Comune Torella dei Lombardi, in rappresentanza dell'Unione dei Comuni dell'alta Irpinia, ha contribuito con specifiche conoscenze inerenti agli assetti organizzativi/gestionali delle PA di livello comunale che hanno consentito la messa a punto di un efficace Sistema di reciproco scambio dei dati tra Enti Regionali e Comuni. L'apertura in modalità SaaS di I.TER. ad Enti Locali Pilota ha permesso ai partner di progetto di utilizzare evolute piattaforme di WEB-GIS e SSD per la creazione di cartografia e reportistica avanzata e, al contempo ha consentito agli enti regionali di semplificare le procedure di acquisizione e omogeneizzazione dei dati trasmessi dai Comuni finalizzati alla redazione di Piani e Programmi Regionali. Il Sistema, applicabile in altri contesti comunali, può rappresentare un grande supporto all'operato dei Comuni che operano in un contesto di ristrettezza economica. Un'ulteriore azione sperimentale è stata dedicata alla diffusione d'uso dei moduli di Geo-Community e Apps Mobile che hanno permesso ai cittadini di comunicare con le PA su tematiche di interesse pubblico veicolando dati georeferenziati direttamente agli uffici competenti.

Il Progetto ha permesso, nei contesti organizzativi-operativi delle Regioni Toscana e Molise e del Comune di Torella dei Lombardi, di utilizzare il Modello di Gestione Integrata delle Entità e degli Eventi territoriali e del relativo Geographic Cloud I.TER. di Regione Campania come buona pratica. Le Strutture competenti in ICT dei Partner hanno collaborato alle attività di installazione di I.TER. presso un Data Center condiviso e sua successiva erogazione ai partner in modalità di Cloud Computing. Sono state individuate all'interno dei partner regionali riusanti delle Direzioni pilota, che successivamente al trasferimento del Know How organizzativo/tecnico e la messa a sistema dei data base di interesse, hanno applicato il Modello di gestione di eventi territoriali di natura trasversale e di interesse strategico (Rete Natura 2000, rischio sismico,gestione dei rifiuti,bonifica dei siti inquinati,Piani di Emergenza Ambientali,risorse energetiche,agricoltura,sviluppo rurale,etc). La sfida affrontata con questa Strategia si è concretizzata nella creazione di una rete di cooperazione interregionale, intraregionale e interistituzionale che ha collaborato alla gestione e allo sviluppo della

buona pratica apportando concrete evoluzioni al modello originale. L'evoluzione della buona pratica ha previsto la diffusione di Geo-Community/Apps Mobile per favorire i meccanismi di cittadinanza digitale e ha messo a punto, anche attraverso il contributo del Comune partner Torella dei Lombardi, una collaborazione Regione /Comuni per il reciproco scambio di dati per affinare la conoscenza del territorio

Il Progetto I.N.TER.PA si è sviluppato non soltanto a livello orizzontale rafforzando la cooperazione tra gli Enti Regionali del partenariato ma anche a livello verticale coinvolgendo la cittadinanza, gli enti territoriali (grazie al contributo del PArtner Comunale) e gli stakeholders quali nodi attivi della rete con il doppio ruolo di produttori e fruitori del dato. La conoscenza territoriale consente di rafforzare la capacità valutativa delle PA e sta alla base di azioni di pianificazione e gestione che si rivelino realmente efficienti ed efficaci e in grado di incidere significativamente sullo sviluppo socio-economico del territorio. Così, l'aumento del grado di previsionalità dei fenomeni e la semplificazione delle attività di individuazione delle criticità sarà in grado di aumentare la velocità e la capacità di risposta delle PA e, permetterà l'individuazione delle azioni prioritarie di intervento, indirizzando in modo mirato gli investimenti necessari alla risoluzione delle emergenze

5 SINTESI DELLA BUONA PRATICA

La Regione Campania ha avviato negli ultimi anni un ambizioso progetto di innovazione e sviluppo di tecnologie digitali che ha portato nel 2015 alla realizzazione della piattaforma “i.TER Campania - L'anagrafe delle Entità e degli Eventi Territoriali”.

i.TER è ad oggi la piattaforma di cloud geografico regionale e consente l’acquisizione, la creazione, la metadatazione, la condivisione e l’analisi di dati geografici.

i.TER funge da collettore per l’intera base della conoscenza geografica regionale, garantendo al contempo l'interoperabilità, attraverso servizi standard, con il patrimonio informatvo nazionale e internazionale.

La piattaforma i.TER Campania rappresenta un caso di eccellenza nel panorama nazionale, ottenendo nel 2016 il premio Agende Digitali Regionali ed è ad oggi utilizzata con successo dalle diverse Direzioni Generali della Regione Campania per semplificare i workflow di gestone della conoscenza territoriale e pubblicare le informazioni geografiche regionali in logica open data.

A seguito della suddetta esperienza le Regioni Toscana, Campania e Molise ed il Comune di Torella dei Lombardi (AV) aderiscono al progetto “Network delle Informazioni Territoriali per le Pubbliche Amministrazioni” (I.N.TER.PA).

L'obiettivo del progetto è il trasferimento tecnologico, organizzativo e di governance della piattaforma i.TER Campania alle Pubbliche Amministrazioni aderenti al partenariato.

Il progetto si propone di diffondere la buona pratica implementata in Regione Campania, replicandone i benefici in Regione Toscana, Molise, e presso il Comune di Torella dei Lombardi, creando una rete di cooperazione inter/intraregionale.

Il progetto rappresenta inoltre il punto di partenza per una collaborazione duratura che garantisca uno sviluppo partecipato e condiviso del modello di e-Gov per acquisire economie di scala e per evitare duplicazione di costi.

Le Amministrazioni aderenti al partenariato intendono capitalizzare le opportunità derivanti dalla partecipazione al progetto I.N.TER.PA., mediante lo sviluppo di una propria progettualità che, a partire dal trasferimento tecnologico, organizzativo e di governance della piattaforma i.TER, punta a migliorare i servizi verso i cittadini, gestire in modo efficiente le risorse naturali ed incrementare la

trasparenza della macchina amministrativa.

Di seguito si riportano il dettaglio, per ciascun ente aderente, delle funzionalità adottate

5.1 Regione Toscana

La Toscana è una regione italiana a statuto ordinario di 3.743.370 di abitanti e quasi il 10% del territorio regionale fa parte del sistema di aree naturali protette con una superficie complessiva di circa 227 mila ettari.

Il sistema di aree naturali protette e, più in generale, il sistema ambientale toscano costituiscono un valore da preservare, rappresentando inoltre un importante volano di sviluppo per settori strategici dell'economia regionale, quali il turismo ed il settore agro-alimentare.

Con la legge regionale 22/2015, la Regione Toscana ha preso il posto delle province in tutte le funzioni in materia di ambiente. Dal primo gennaio 2016 è quindi la Regione a rilasciare le autorizzazioni in materia di emissioni in atmosfera, rifiuti, bonifiche ed energia. Inoltre, sono rilasciate dalla Regione le Autorizzazione Integrata Ambientale (AIA) e l'Autorizzazione Unica Ambientale (AUA). Oltre a queste funzioni la Regione ha riacquisito anche le competenze in materia di parchi e biodiversità, divenendo quindi titolare della gestione delle aree protette e della Valutazione di Incidenza.

La Regione Toscana affianca quindi alle tradizionali funzioni di normazione, programmazione, indirizzo e controllo in materia ambientale anche quella autorizzativa, chiudendo quindi il cerchio di un ideale percorso che partendo dalla legge termina al rilascio di una autorizzazione.

In tale contesto, la Regione Toscana intende portare avanti tre interventi il cui fine comune è supportare i diversi attori regionali nella condivisione dei dati ambientali e nello sviluppo di politiche di salvaguardia e valorizzazione del patrimonio ambientale:

- Mappatura delle autorizzazioni ambientali;
- Definizione di un modello di gestone dei dati delle riserve naturali;

I suddetti progetti hanno inoltre il fine di incrementare il rapporto di fiducia tra cittadini ed amministrazione, attraverso una migliore comunicazione al pubblico delle azioni intraprese da Regione Toscana in materia ambientale.

Per il conseguimento dei suddetti obiettivi, la Regione richiede un mix coordinato di attività di supporto specialistico e sviluppo di nuove funzionalità della piattaforma i.TER Campania.

Il referente per la Regione Toscana è Sauro del Turco

5.2 Regione Molise

Il Molise è una regione italiana a statuto ordinario di 308.696 di abitanti il cui territorio regionale è caratterizzato da ampie aree naturali protette e vaste zone boschive con importanti varietà di fauna e flora.

In tale contesto, la Regione Molise intende portare avanti tre interventi in sinergia con gli analoghi interventi della Toscana e di Torella dei Lombardi:

- Mappatura delle autorizzazioni ambientali;
- Definizione di un modello di gestone dei dat delle riserve naturali;
- Sperimentazione di una App sui servizi locali che visualizzi i dati provenienti dai dispositivi OBU installati da Regione Molise in aderenza a quanto previsto per il comune di Torella dei Lombardi.

Il referente per la Regione Molise è ???

5.3 Comune Torella dei Lombardi

Torella dei Lombardi è un comune italiano di 2.240 abitanti della provincia di Avellino in Campania. Il territorio comunale è caratterizzato da un'orografia collinare complessa e da un tessuto insediativo altamente disperso.

Durante i mesi invernali, il Comune è soggetto a frequenti e copiose nevicate che possono costituire un serio impedimento per la mobilità dei cittadini. Tali circostanze rendono i servizi comunali di pulizia e sgombero neve ed i servizi di scuolabus dei servizi di primaria importanza per il Comune ed i suoi cittadini: una mancata o parziale erogazione dei medesimi può infatti comportare seri problemi alla popolazione residente ed in particolar modo agli anziani, alle persone con mobilità ridotta ed alla popolazione residente in zone isolate o poco connesse del territorio comunale.

I servizi sono ad oggi espletati da società esterne al Comune ed i contratti di erogazione dei servizi prevedono il soddisfacimento di alcuni SLA contrattuali.

Il Comune di Torella intende dotare i mezzi comunali che concorrono all'espletamento di tali servizi di dispositivi OBU in grado di inviare la posizione del veicolo, mediante tecnologia GPS.

Al fine di migliorare la qualità dei suddetti servizi, razionalizzare i cost delle forniture e offrire ai cittadini risposte sempre più efficienti e tempestive, l'Amministrazione Comunale richiede lo sviluppo di soluzioni tecnologiche per il monitoraggio e il controllo delle flotte dei veicoli impiegati nei servizi di scuolabus e nei servizi di pulizia e sgombero della neve.

Nello specifico, l'Amministrazione richiede di sviluppare ed integrare nella piattaforma i.TER le tecnologie necessarie a:

- Ricevere ed archiviare i dati provenienti dalle OBU installate sui mezzi;
- Interrogare tali dati mediante una interfaccia utente semplice e intuitiva;
- Verificare gli SLA contrattualizzati con i fornitori di servizio;
- Produrre reportistica relativa ai servizi in oggetto.

Il Comune richiede inoltre la realizzazione di un POC webApp (di seguito nominata App "Servizi Comunali") che consenta ai propri cittadini di poter:

- Conoscere in tempo reale la posizione dei mezzi spazzaneve e scuolabus;
- Ricevere notifiche rilevanti sul servizio;
- Ricevere informazioni sul tempo di arrivo stimato degli scuola bus;
- Ricevere informazioni sull'espletamento dei servizi di rimozione della neve.

L'App costituirà il punto di partenza per la semplificazione del rapporto tra Amministrazione e cittadini e dovrà essere composta di due sezioni: una per i servizi di scuolabus ed una per quelli di spazzaneve e dovrà essere progettata in maniera tale da poter costituire in futuro la base sulla quale poter aggiungere nuovi servizi.

Il referente per il Comune di Torella dei Lombardi è ???

6 GESTIONE DEL TRASFERIMENTO E ADOZIONE DELLA BUONA PRATICA

Il Progetto si propone di diffondere nei contesti organizzativi-operativi delle Regioni Toscana e Molise e del Comune di Torella dei Lombardi l'utilizzo del Modello di Gestione Integrata delle Entità e degli Eventi territoriali e del relativo Geographic Cloud I.TER. già realizzato dalla Regione Campania come buona pratica.

La strategia prevede la collaborazione tra i partner per l'analisi degli aspetti gestionali/organizzativi/tecnologici/amministrativi/informativi delle azioni di ricerca, selezione, trasferimento, adozione e gestione a regime della buona pratica.

Le Strutture competenti in ICT dei Partner collaboreranno alle attività di installazione di I.TER. presso un Data Center condiviso e sua successiva erogazione ai partner in modalità di Cloud Computing. Verranno individuate all'interno dei partner regionali riusanti Direzioni pilota, che successivamente, al trasferimento del Know How organizzativo/tecnico e la messa a sistema dei data base di interesse, applicheranno il Modello alla gestione di eventi territoriali di natura trasversale e di interesse strategico (Rete Natura 2000, rischio sismico, gestione dei rifiuti, bonifica dei siti inquinati, Piani di Emergenza Ambientali, risorse energetiche, agricoltura, sviluppo rurale,etc).

La sfida che si intende affrontare con questa Strategia è quella di creare una rete di cooperazione interregionale, intraregionale e interistituzionale che collabori alla gestione e allo sviluppo della buona pratica apportando concrete evoluzioni al modello originale. L'evoluzione della buona pratica intende diffondere l'uso di Geo-Community/Apps Mobile per favorire i meccanismi di cittadinanza digitale e mettere a punto, anche attraverso il contributo del Comune partner Torella dei Lombardi, una collaborazione Regione /Comuni per il reciproco scambio di dati per affinare la conoscenza del territorio.

Il Progetto I.N.TER.PA determinerà un significativo miglioramento dell'efficacia e dell'efficienza della procedura della PA attraverso l'omogeneizzazione e la standardizzazione su più livelli delle metodologie e delle strumentazioni in uso.

L'aumento della capacità di diffusione di dati delle PA e il coinvolgimento diretto dei cittadini e degli stakeholders secondo l'ottica di inclusione digitale migliorerà la trasparenza, la partecipazione e la comunicazione a sostegno dell'azione amministrativa.

Nell'ottica di rendere il modello del Riuso applicato nel progetto I.N.TER.PA verranno organizzati 2 Eventi per singola regione che avranno lo scopo di divulgare la Buona Pratica applicata e la diffusione degli elementi di miglioramento.

Inoltre, le caratteristiche tecnologiche della piattaforma I.TER., che ne assicurano la sua piena interoperabilità con le banche interne ed esterne agli uffici regionali, permetteranno il convogliamento dei dati di diversa natura e origine su un unico Data Hub. L'analisi integrata dei dati restituirà un'adeguata visione d'insieme dei fenomeni in atto che potranno essere correlati secondo una logica causa-effetto

6.1 Piano di Adozione della Buona Pratica

Il progetto prevede pertanto l'implementazione delle seguenti macro - attività:

Attività	Azione I.N.TER.PA
Assessment e progettazione	Azione 2 e Azione 3
Realizzazione di un'architettura applicativa IoT I.N.TER.PA	Azione 2 e Azione 3
Progettazione e realizzazione di un'applicazione WEB GIS – Fleet Management	Azione 3
Progettazione e realizzazione di un'applicazione WEB GIS – Servizi Informazioni Meteo	Azione 4
Progettazione e realizzazione del verticale WEB GIS on i.TER – Gestione Riserve Naturali	Azione 4
Progettazione e realizzazione del verticale WEB GIS on i.TER – Autorizzazioni Ambientali	Azione 4
Progettazione e realizzazione di un APP Mobile.	Azione 4
Comunicazione	Azione 5

Realizzazione Architettura Applicativa IoT I.N.TER.PA	La realizzazione del caso d'uso di Monitoraggio e Controllo della Flotta, si avvarrà di una soluzione già sviluppata da Almaviva nell'ambito del progetto i.TER Campania, ovvero una Piattaforma IoT; questa sarà di riferimento anche per il progetto in essere INTERPA. Nell'ambito del progetto INTERPA tale soluzione consente di: ■ Acquisire i dati di posizione dalle OBU, presenti sui mezzi spazzaneve e scuolabus, mediante protocollo MQTT ■ Memorizzare i dati acquisiti su diverse tecnologie di persistenza; ■ Elaborare i dati acquisiti attraverso strumenti di processing e machine learning; ■ Visualizzare i dati acquisiti ed i dati elaborati attraverso una interfaccia web; ■ Cooperare con sistemi di terze parti esponendo o interrogando dati;	**Progettazione e realizzazione di un'applicazione WEB GIS – Servizi Informazioni Meteo**	Progettazione e Realizzazione di un servizio WEB GIS di servizio meteo che contiene informazioni previsionali raccolte da servizi meteo esterni disponibili online. A tal proposito si utilizzano i dati darksky.net, i quali sono appunto esposti tramite chiamate API REST. Il servizio è utilizzato nella modalità free, sono previsti così un massimo di 1000 chiamate gratuite giornaliere. Tale numero è così ripartito: ■ Le chiamate verranno eseguite ogni giorno esclusivamente alle ore 00:00. ■ Sarà richiesta la previsione per le ore 8:00, 16:00, 24:00. ■ Il totale di celle previsionali contigue vanno a coprire l'intera area di interesse. In output sono forniti degli alert costruiti a partire dalle previsioni e dal superamento di soglie limite predefinite, soglie adeguate alle specifiche variabili meteo: pioggia,

	Costruire reportistica sui dati acquisiti ed elaborati		vento, grandine, neve,temperatura. Le soglie fanno riferimento a valori tipici in uso alle agenzie del territorio.
Progettazione e realizzazione di un'applicazione WEB GIS – Fleet Management	Progettazione e realizzazione di un'applicazione WEB GIS di Fleet management che si poggiano sull'acquisizione dei dati rilevati tramite GPS - OBU (on boarding unit) e sulle primitive dell'engine i.TER, in particolare: ■ Inserimento Device ■ Disabilitazione Device ■ Inserimento Veicolo ■ Abilitazione Veicolo ■ Associazione Veicolo con Device (OBU) ■ Disabilitazione Veicolo ■ Inserimento Operatore ■ ….	**Progettazione e realizzazione del verticale WEB GIS on i.TER – Gestione Riserve Naturali**	Progettazione e Realizzazione di un verticale WEB GIS on i.TER per la Gestione delle Riserve Naturali nell'ottica di fornire alla Regione Toscana uno strumento di WebMapping che sia di supporto ai processi interpretativi e decisionali dell'Amministrazione si procederà con la realizzazione di: ■ Database Geografico per la accolta di dati che consentiranno il monitoraggio ambientale e la produzione di shape associati ad info alfanumeriche ■ Realizzazione di una pagina WEB GIS con area pubblica e privata. L'area pubblica consentirà alla Regione di avere una vetrina di tutti gli eventi che saranno ricercabili sia tramite ricerca alfanumerica che geografica.
Progettazione e realizzazione del verticale WEB GIS on i.TER – Autorizzazioni Ambientali	Progettazione e Realizzazione di un verticale WEB GIS on i.TER per le Autorizzazioni Ambientali che consentirà il caricamento delle Autorizzazioni Ambientali attraverso la creazione di un modello dati per l'integrazione dei dati geografici e di presentazione riguardanti le autorizzazioni ambientali. Il modello va ad uniformare, inoltre, le informazioni di pertinenza; tra cui dati multimediali quali foto e video. I dati sono disponibili (compresi quelli multimediali) alla visualizzazione georeferita su mappa tramite operazioni di identiy e querying GIS	**Progettazione e realizzazione di un APP Mobile**	Progettazione e Realizzazione di due applicazioni Mobile da fornire ai cittadini del comune di Torella dei Lombardi e della regione Molise per fornire in tempo reale lo stato del servizio dei mezzi spazzaneve e la posizione degli scuolabus in arrivo. Le app verranno realizzate in ottica di customizzazione per i due territori destinatari e dovranno erovare le seguenti funzionalità tramite colloquio con API Rest Json messe a disposizione dai sistemi di BackEnd. • Dashboard: l'applicazione consentirà all'utente di ottenere rapidamente informazioni in real time sullo stato dei servizi di Scuola Bus e mezzi Spazza neve. Potrà inoltre ottenere informazioni sul meteo basate sulla propria posizione • Scuola bus: l'utente potrà visionare su mappa i percorsi previsti per gli scuola bus del territorio e verificare i tempi di arrivo real time venendo a conoscenza della posizione corrente del singolo bus di

			interesse • Spazza neve: l'utente potrà visionare su mappa i percorsi previsti per i mezzi di spazzamento neve sul territorio

Di seguito riportiamo il piano di dettaglio di ciascun task sopra descritto, con il relativo stato di avanzamento:

Attività	% Avanzamento	Inizio	Fine
ITER RIUSO Molise Toscana	**4%**	**15-lug-19**	**06-feb-20**
WEB GIS	**9%**	**15-lug-19**	**04-nov-19**
Assessment e Progettazione	**33%**	**15-lug-19**	**23-ago-19**
Installazione e configurazione delle componenti	**0%**	**26-ago-19**	**11-set-19**
Installazioni componenti GIS	0%	26-ago-19	06-set-19
Predisposizione del DB	0%	09-set-19	11-set-19
WebGIS App - Fleet management	**0%**	**26-ago-19**	**04-ott-19**
Sviluppo calcolo delle distanze percorse per veicolo	0%	26-ago-19	06-set-19
Sviluppo riepilogo delle Zone Coperte	0%	09-set-19	13-set-19
Sviluppo SLA di servizio	0%	16-set-19	20-set-19
Test	0%	23-set-19	02-ott-19
Documentazione	0%	03-ott-19	04-ott-19
WebGIS App - Servizio informazioni meteo	**0%**	**26-ago-19**	**30-set-19**
Sviluppo	0%	26-ago-19	02-set-19
Test	0%	23-set-19	26-set-19
Documentazione	0%	27-set-19	30-set-19
Verticale WebGIS on i.TER – Gestione Riserve Naturali	**0%**	**07-ott-19**	**04-nov-19**
Sviluppo	0%	07-ott-19	25-ott-19
Test	0%	28-ott-19	31-ott-19
Documentazione	0%	01-nov-19	04-nov-19
Verticale WebGIS on i.TER – Autorizzazioni Ambientali	**0%**	**26-ago-19**	**30-set-19**
Sviluppo	0%	26-ago-19	29-ago-19
Test	0%	23-set-19	26-set-19
Documentazione	0%	27-set-19	30-set-19
IOT	**0%**	**07-ago-19**	**05-nov-19**
Assessment e Progettazione	**0%**	**07-ago-19**	**23-ago-19**
Installazione e configurazione delle componenti	**0%**	**26-ago-19**	**13-set-19**
Installazioni componenti IoT	0%	26-ago-19	04-set-19
Predisposizione utenze ACL	0%	05-set-19	10-set-19
Predisposizione del DB	0%	11-set-19	13-set-19
Sviluppo	**0%**	**16-set-19**	**22-ott-19**
Persistenza messaggi OBU	0%	16-set-19	20-set-19
Pubblicazione posizione mezzi in tempo reale	0%	23-set-19	27-set-19
Predisposizione template geofencing	0%	30-set-19	08-ott-19
Sviluppo REST API	0%	09-ott-19	22-ott-19
Test	**0%**	**23-ott-19**	**30-ott-19**
Documentazione	0%	31-ott-19	05-nov-19
Stesura Progetto Esecutivo (WEB GIS + IOT)	**0%**	**30-set-19**	**30-set-19**

Attività	% Avanzamento	Inizio	Fine
App Mobile	**0%**	**09-ott-19**	**22-gen-20**
Progettazione	**0%**	**09-ott-19**	**21-nov-19**
Mappa di navigazione	0%	09-ott-19	22-ott-19
Approvazione Mappa	0%	22-ott-19	22-ott-19
Sketch UX	0%	23-ott-19	11-nov-19
Approvazione Sketch UX	0%	11-nov-19	11-nov-19
Proposta UI	0%	12-nov-19	21-nov-19
Approvazione Proposta UI	0%	21-nov-19	21-nov-19
Integrazione API	**0%**	**22-nov-19**	**13-dic-19**
Integrazione Logica di business	0%	16-dic-19	10-gen-20
Testing	0%	13-gen-20	22-gen-20
Collaudo	**0%**	**23-gen-20**	**30-gen-20**
Preparazione al Go-live	**0%**	**31-gen-20**	**05-feb-20**
Go-live	**0%**	**06-feb-20**	**06-feb-20**

6.2 MANSIONARIO

Nominativo	Qualifica	in INTERPA
Sauro Del Turco	Funzionario Informatico	
Laura Pacini	Funzionario di programmazione	
Marco Giovannetti	Funzionario amministrativo esperto	
Simone Secci	Funzionario Informatico	
Elettra Iannone	Assistente Amministrativo	
Massimo Sernesi	Funzionario Informatico	
Alessandro Tarchi	Funzionario sistemi informativi e tecnologie	
Alessandro Pini	Assistente sistemi informativi e tecnologie	
Vincenzo Martiello	Funzionario sistemi informativi e tecnologie	

Nominativo	Qualifica	in INTERPA
Paolo Matteini	Funzionario sistemi informativi e tecnologie	
David Tei	Funzionario di programmazione	
Ilaria d'Urso	Funzionario di programmazione	
Anna Iengo	Assistente tecnico – professionale	
Cristiana Natali	Assistente tecnico – professionale	

7 GESTIONE ORGANIZZATIVA DEL TRASFERIMENTO DELLA BUONA PRATICA

Di seguito si riportano la descrizione di quanto verrà attuato per il trasferimento della Buona Pratica

7.1 MONITORAGGIO E CONTROLLO FLOTTA SCUOLABUS E SPAZZANEVE

Il Comune di Torella dei Lombardi e la Regione Molise intendono migliorare la qualità dei servizi comunali quali:

- Pulizia e sgombero neve
- Scuolabus

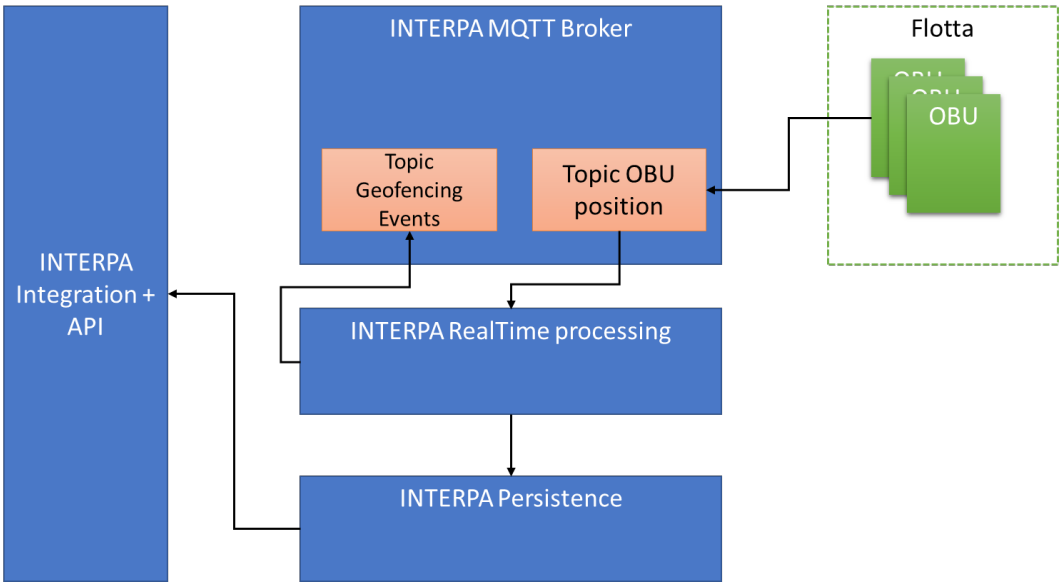
Per raggiungere tale obiettivo, i mezzi delle flotte scuolabus e spazzaneve verranno dotati di dispositivi OBU in grado di inviare la posizione del veicolo, mediante tecnologia GPS. Contestualmente verrà predisposto un flusso di acquisizione dei dati provenienti dai dispositivi OBU installati sulle flotte presenti sul territorio.

Lo scopo di tale processo è quello di acquisire i dati di posizione tramite un Message Broker di piattaforma su protocollo MQTT, analizzarli al fine del calcolo dei livelli di servizio, memorizzarli e allo stesso tempo renderli disponibili mediante API REST.

Si descrivono di seguito i flussi principali per espletare i casi d'uso richiesti.

7.1.1 Acquisizione messaggi OBU

Le OBU installate sui mezzi, secondo logiche prestabilite, campionano la propria posizione e una serie di altre informazioni impacchettando il tutto all'interno di un payload di dati. Il payload viene poi trasmesso alla piattaforma INTERPA tramite il protocollo MQTT su rete mobile.



Il protocollo MQTT che è un O.A.S.I.S standard normato ISO/IEC 20922:2016 per l'IoT date le sue caratteristiche ed è di tipo asincrono secondo lo schema publish/subscribe. Le OBU dovranno essere predisposte per comunicare mediante tale protocollo, in particolare secondo la specifica 3.1.1, in modo da supportare tutti i livelli di qualità del servizio (0,1,2).

In particolare, i livelli di servizio sono di seguito descritti:

- QoS 0 – at most once: gestisce il "best effort delivery". È detto anche fire & forget e non contempla la conferma della ricezione, assicurando il medesimo livello di servizio del trasporto TCP/IP.
- QoS 1 – at least once: garantisce la consegna del messaggio almeno una volta verso un ricevente; il messaggio può tuttavia essere ricevuto anche più volte.
- QoS 2 – exactly once: garantisce la consegna del messaggio una e una sola volta verso un ricevente. Ogni dispositivo dovrà essere in grado di presentarsi alla piattaforma con un clientId univoco in modo da non generare conflitti.

Dal punto di vista della configurazione, le OBU dovranno fornire le seguenti funzionalità:

- Sicurezza: le centraline devono essere in grado di dialogare con la piattaforma sia in chiaro che in SSL ma, a prescindere dalla sicurezza del canale nella sottoscrizione o pubblicazione verso la piattaforma, devono essere in grado di inserire le credenziali necessarie per l'autenticazione. Per la comunicazione in SSL le centraline dovranno avere la possibilità di inserire un certificato emesso dal fornitore della piattaforma in modo da poterlo utilizzare nell'autenticare il broker (canale trasmissivo su cui vengono sottoscritti i messaggi MQTT dal campo).
- Tipologia di comunicazione: le centraline dovranno essere in grado di abilitare il meccanismo

di pubblicazione (specificando QoS, ClientId e CleanSession) e sottoscrizione (specificando QoS, ClientId e CleanSession) per scambiare messaggi tramite il Message Broker messo a disposizione dalla piattaforma. Qualora fosse necessario, tramite il meccanismo della sottoscrizione, le centraline potranno ricevere comandi dalla piattaforma al fine di effettuare attuazioni. Le centraline devono essere in grado di inviare messaggi almeno in uno dei due formati standard previsti: XML e JSON; il JSON è tuttavia preferibile per la sua minore verbosità.

- Struttura dei messaggi: la struttura dei messaggi deve essere definita in fase preliminare e precede la fase di instaurazione della comunicazione con la piattaforma; questo è necessario per permettere alla stessa di elaborare il contenuto informativo in maniera adeguata. Una volta definita la struttura, le centraline dovranno inviare esattamente il messaggio concordato.

7.1.2 Modulo verifica SLA servizi Sgombero Neve e Scuolabus

A partire dai messaggi inviate dalle OBU sui mezzi spazzaneve e scuolabus, la piattaforma effettua delle logiche di elaborazione finalizzate al calcolo degli SLA contrattuali. Nello specifico, le componenti della piattaforma possono effettuare dei calcoli sui dati grezzi arricchendoli di informazioni e possono inoltre analizzarli al fine di individuare particolari condizioni e sollevare contestualmente degli alert sottoforma di messaggi MQTT.

La Piattaforma espone diverse interfacce al fine di:

- Acquisire messaggi di posizione
- Creare, aggiornare e cancellare regole di geofencing
- Recuperare la lista delle regole di geofencing definite
- Interrogare lo storico delle posizioni dei mezzi
- Recuperare informazioni derivanti da aggregazioni di tipo statistico, nello specifico:
    - La distanza media (espressa in metri) percorsa giornalmente in tutto il range temporale specificato
    - La distanza totale (espressa in metri) percorsa per singolo giorno
    - La velocità media (espressa in Km/h)

7.1.2.1 Regole di Geofencing

Il Complex Event Processor di Piattaforma è la componente deputata alla creazione e gestione delle regole di geofencing. Le regole di geofencing sono quelle regole che si avvalgono della posizione GPS per scatenare eventuali alert sui dati.

La Piattaforma è in grado di supportare le tre seguenti tipologie di regole:

- Entrata in una zona geografica predefinita: il CEP solleva uno specifico alert quando un veicolo entra in una zona di interesse
- Uscita da una zona geografica predefinita: il CEP solleva uno specifico alert quando un veicolo esce da una zona di interesse
- Sosta in una zona geografica predefinita: il CEP solleva uno specifico alert quando un veicolo rimane fermo per più di un certo tempo all'interno della zona di interesse

7.2 Advanced Dashboard Analytics

L'Advanced Dashboard Analytics è lo strumento di reportistica del progetto INTERPA. Questo strumento, direttamente tramite la sua interfaccia grafica, è in grado di:

- Connettersi a differenti data source sia interni che esterni
- Effettuare QUERY in linguaggio sql-like tramite un SQL editor web
- Rappresentare il risultato delle estrazioni dei dati in forma grafica
- Filtrare i dati rappresentati attraverso filtri logici
- Allestire Dashboard multi-contesto in modalità drag-and-drop
- Gestire i permessi sulle visualizzazioni delle varie dashboard

Tra le altre funzionalità principali è in grado di:

- condividere le Query
  - condividere gli sql trai vari utenti
  - riusare le query degli altri utenti
- pianificare le Query
  - configurare l'intervallo temporale di refresh
  - configurare l'esecuzione in determinati giorni o eventi
- fare caching sui dati estratti

Definita la connessione ad un determinate Dataset il flusso di lavoro dell'utente è così definibile:

- Scrivere la query attraverso il query editor
- Creare il grafico e configurarlo in base al tipo di rappresentazione
- Esportare I dati del grafico in csv ed esportare l'immagine del grafico in formato JPG/PNG.
- Inserire il grafico all'interno delle dashboard tematiche

L'Advanced Dashboard Analytics inoltre offre la possibilità di essere ulteriormente esteso nelle seguenti modalità operative:

- La programmazione di parametri e filtri logici
- L'interoperabilità con altri sistemi grazie alle API documentate
- Customizzazioni specifiche attraverso il codice sorgente open source

L'Advanced Dashboard Analytics offre inoltre la possibilità di gestire la profilazione e i permessi degli utenti attraverso:

- Creazione e gestione di gruppi
- Accesso a determinati dataset per utenti e gruppi
- Accesso alle dashboard per determinati utenti e gruppi

7.3 WebGIS App - Fleet management

Le capabilities di Fleet management macro-funzionali offerte si poggiano sull'acquisizione dei dati rilevati tramite GPS - OBU (on boarding unit) e sulle primitive dell'engine i.TER, in particolare:

- Inserimento Device
- Disabilitazione Device
- Inserimento Veicolo
- Abilitazione Veicolo
- Associazione Veicolo con Device (OBU)
- Disabilitazione Veicolo
- Inserimento Operatore
- Associazione Operatore con Device (Mobile)
- Inserimento Flotta
- Inserimento Missione
- Assegnazione Flotta a Missione
- Inserimento Regola di Allerta per Missione
- Pannello Di Alerts
- Alerts Sconfinamento
- Filtro per Flotte
- Filtro per Missione
- Filtro per Veicolo
- Storico Percorsi
- Colloquio Apparati
- Anagrafiche Dati
- Dati Monitoraggio
- OGC Compliance (REST API)
- REST API Anagrafiche
- REST API Monitoraggio
- Invio Posizione Mobile
- Buffering in Assenza di Rete
- Regole di Geofencing

7.3.1 Il calcolo delle distanze percorse per veicolo

Lo strumento - a partire dalle associazioni tra veicolo, dispositivo GPS, assegnazione a flotta (ovvero un gruppo di veicoli classificanti dei gestori o dei servizi come gli scuolabus e gli sgombraneve) - cumula le distanze lineari tra un rilievo puntuale ed il successivo, il dato così ottenuto viene storicizzato e disponibile per il querying e la reportistica.

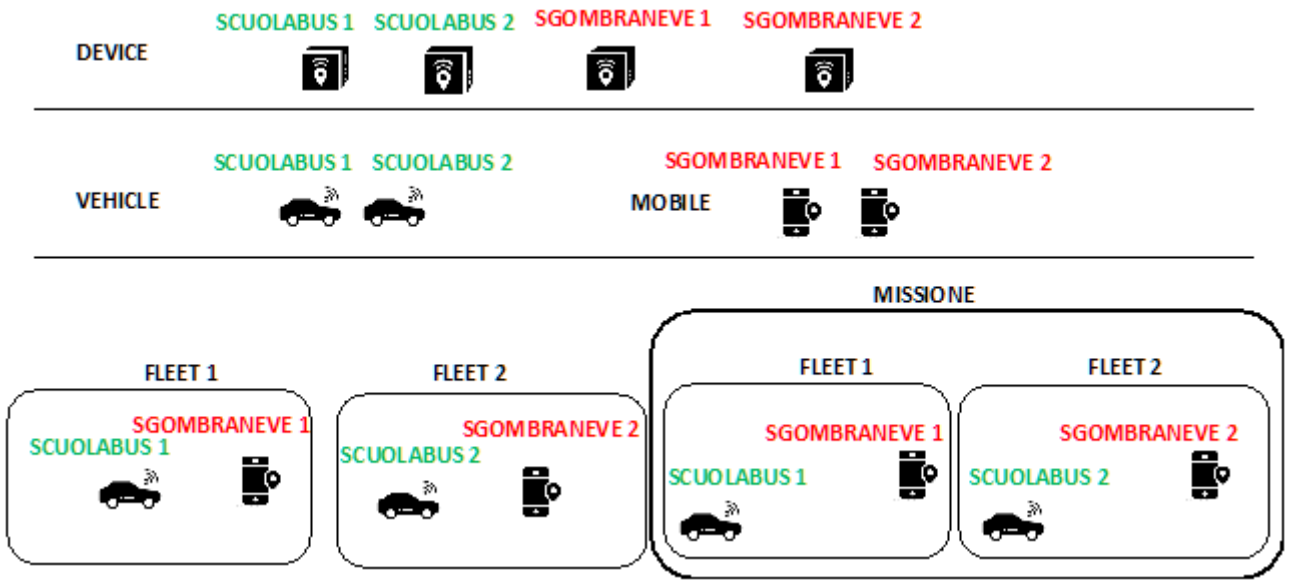


Figura 1. Riepilogo degli elementi e funzioni strutturali

7.3.2 Riepilogo delle Zone Coperte

Attraverso l'impostazione di regole di geofencing (come l'ingresso e uscita del veicolo da area di interesse) e lo storage dei relativi eventi che occorrono è offerto uno strumento capace di riportare il calcolo percentuale delle aree servite su quelle totali, distribuendone l'output su periodi temporali.

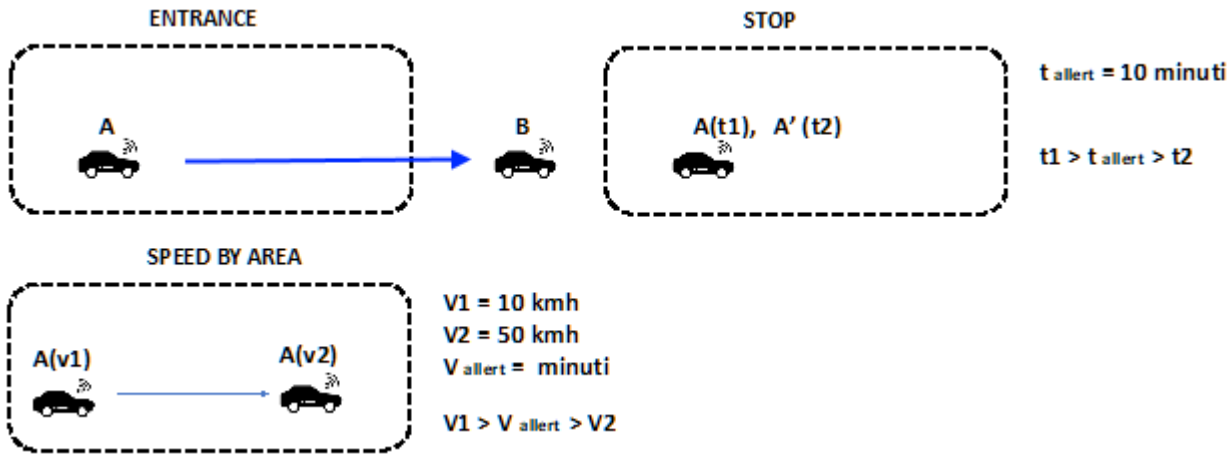


Figura 2. Regole di Geofencing utili ad individuare aree servite.

7.3.3 SLA di servizio

Lo strumento consente di inserire le regole SLA inerenti all'allontanamento per un certo periodo da una zona segnata a priori. Attraverso questa operatività di rimando può essere evidenziato l'allontanamento di un veicolo da un percorso areale prima delimitato.

7.4 WebGIS App - Servizio informazioni meteo

Il servizio meteo contiene informazioni previsionali raccolte da servizi meteo esterni disponibili online.
A tal proposito si utilizzano i dati darksky.net, i quali sono appunto esposti tramite chiamate API REST.
Il servizio è utilizzato nella modalità free, sono previsti così un massimo di 1000 chiamate gratuite giornaliere.
Tale numero è così ripartito:

- Le chiamate verranno eseguite ogni giorno esclusivamente alle ore 00:00.
- Sarà richiesta la previsione per le ore 8:00, 16:00, 24:00.
- Il totale di celle previsionali contigue vanno a coprire l'intera area di interesse.

In output sono forniti degli alert costruiti a partire dalle previsioni e dal superamento di soglie limite predefinite, soglie adeguate alle specifiche variabili meteo: pioggia, vento, grandine, neve,temperatura.
Le soglie fanno riferimento a valori tipici in uso alle agenzie del territorio.

7.5 Verticale WebGIS on i.TER – Autorizzazioni Ambientali

La gestione del caricamento delle Autorizzazioni Ambientali viene espletata tramite un verticale dotato di strumenti specifici per l'uso di scopo. E' creato un modello dati per integrare i dati geografici e di presentazione riguardanti le autorizzazioni ambientali. Il modello va ad uniformare, inoltre, le informazioni di pertinenza; tra cui dati multimediali quali foto e video. I dati sono disponibili (compresi quelli multimediali) alla visualizzazione georeferita su mappa tramite operazioni di identiy e querying GIS.

7.6 Verticale WebGIS on i.TER – Gestione Riserve Naturali

La gestione delle riserve naturali viene espletata tramite un verticale dotato di strumenti specifici per l'uso di scopo. Nell'ottica di fornire alla Regione Toscana uno strumento di WebMapping che sia di supporto ai processi interpretativi e decisionali dell'Amministrazione si procederà con la realizzazione di quanto di seguito riportato

7.6.1 Realizzazione database geografico

Al fine di di realizzare un database geografico esaustivo allo scopo dell'Amministrazione si raccoglieranno dati di tipo

- Amministrativo, Tecnico, Scientifico come ad esempio normative di riferimento, atti istitutivi, regolamenti, piani di gestione etc:
  Tali datiDovranno essere raccolti e organizzati al fine di ottenere uno shape lineare a cui dovranno essere associate non solo

- Geografico (shape file) riferiti ai suddetti dati
- Informativo, Conoscitivo, Divulgativo; come ad esempio sentieri, file multimediali, pubblicazioni etc…

I dati suddetti che verranno raccolti e organizzati, al fine di rendere esaustivo il database geografico, verranno utilizzati per poter produrre uno shape lineare o puntuale al quale dovranno essere associati sia informazioni di tipo alfanumerico che di tipo multimediale, quali video e/o foto, in alcuni casi si dovrà poter riferire lo shape a link esterni, come nel caso dei monitoraggi ambientali

I confini delle aree protette e tutte le informazioni "ufficiali" verranno reperite tramite un servizio wms, che la Regione Toscana esporrà.

7.6.2 Realizzazione Applicazione Web Gis

Si dovrà prevedere la realizzazione di una pagina WebGis per ciascuna delle seguenti tre macroaree:

- - Informativo
- - Geografico
- - Tecnico – Amministrativo
- - Social

Si forniranno le funzionalità necessarie alla creazione di link interattivi e web necessarie all'inserimento di contenuti multimediali come canale youtube, pubblicazioni on-line, normativa, ecc.

Questo per permettere all'utente del sistema di poter informarsi in maniera semplice diretta e veloce.

Verrà fornita, inoltre, un'area ad accesso autenticato attraverso la quale la Regione, in un primo momento e a seguire gli enti Parchi e i gestori delle riserve naturali a seguire, possano inserire gli eventi che verranno organizzati nei Parchi e nelle Riserve Naturali della Regione Toscana.

L'area di programmazione degli eventi sarà predisposta al fine di inserire dati quali:

- Data o Periodo
- Titolo dell'Evento
- Descrizione dell'Evento
- Indicazione della Riserva o del Parco
- Località
- Contatti
- Foto e Video relative all'Evento
- Coordinate geografiche che consentano la georeferenzazione dell'evento

Verrà fornita una pagina web pubblica che sarà la vetrina di esposizione di tutti gli eventi relativi ai

Parchi e alle Riserve Naturali della Regione Toscana, in tale pagina web gli utenti potranno ricercare gli eventi sia attraverso l'interazione con una mappa che attraverso l'inserimento di campi testuali e temporali.

Dalla pagina web pubblica si potrà accedere ad un'area riservata, che in questa prima fase sarà usufruibile in ugual misura da utenti specializzati della Regione Toscana e dagli Enti Parchi e Riserve Naturali. Si prevederà anche la possibilità di accedere a tale area riservata agli escursionisti e ai visitatori degli Enti Parchi, che potranno accedere a tale area riservata utilizzando i social, quali Google e Facebook. I dati utilizzati per l'accesso da questi utenti non saranno memorizzati ma unicamente utilizzati per consentire a tali utenti l'accesso all'area riservata.

Attraverso l'accesso all'area riservata sarà possibile, al personale della Regione Toscana e degli Enti Parco e Riserve Naturali, l'inserimento dei dati di programmazione degli eventi sia in modalità alfanumerica che attraverso dei markers su una mappa e di inserire la propria esperienza, ai visitatori ed escursionisti, attraverso testo, emoticon, e markers sulle mappe. Si prevederà una fase di validazione dei dati inseriti da parte di un'utente esterno, prima che questi siano salvati sul database geografico e visibili dall'area pubblica.

7.7 Mobile Applications

Nell'ambito del trasferimento della Buona Pratica verranno implementate due applicazioni mobile da fornire ai cittadini del comune di Torella dei Lombardi e della regione Molise per fornire in tempo reale lo stato del servizio dei mezzi spazzaneve e la posizione degli scuolabus in arrivo. Dove non diversamente indicato, dovranno essere realizzate tenendo conto dei seguenti requisiti comuni:

Sistemi operativi mobile supportati

L'applicazione per dispositivi mobili dovrà essere fruibile da smartphone, rese disponibili sugli store Google Play e App Store gratuitamente, tramite l'account del committente. Saranno garantite per le seguenti versioni di sistemi operativi:

- Android da 5.1 a 9.1
- iOS da 11.0 a 12.2

Geolocalizzazione

Le applicazioni, per permettere l'interfacciamento con le mappe o per altre finalità illustrate a seguire, richiederanno all'utente il permesso di abilitare la geolocalizzazione. Verrà specificato in ogni ambito di utilizzo con quali finalità la posizione utente verrà trattata, e se è necessario che il dato venga tracciato o meno.

7.7.1 Requisiti funzionali

Le app verranno realizzate in ottica di customizzazione per i due territori destinatari e dovranno erogare le seguenti funzionalità tramite colloquio con API Rest Json messe a disposizione dai sistemi di BackEnd.

- Dashboard: l'applicazione consentirà all'utente di ottenere rapidamente informazioni in real time sullo stato dei servizi di Scuola Bus e mezzi Spazza neve. Potrà inoltre ottenere informazioni sul meteo basate sulla propria posizione
- Scuola bus: l'utente potrà visionare su mappa i percorsi previsti per gli scuola bus del territorio e verificare i tempi di arrivo real time venendo a conoscenza della posizione corrente del singolo bus di interesse
- Spazza neve: l'utente potrà visionare su mappa i percorsi previsti per i mezzi di spazzamento neve sul territorio

8 GESTIONE DEGLI ASPETTI TECNOLOGICI PER IL TRASFERIMENTO DI ADOZIONE DELLA BUONA PRATICA

L'obiettivo di tale sezione è quello di fornire:

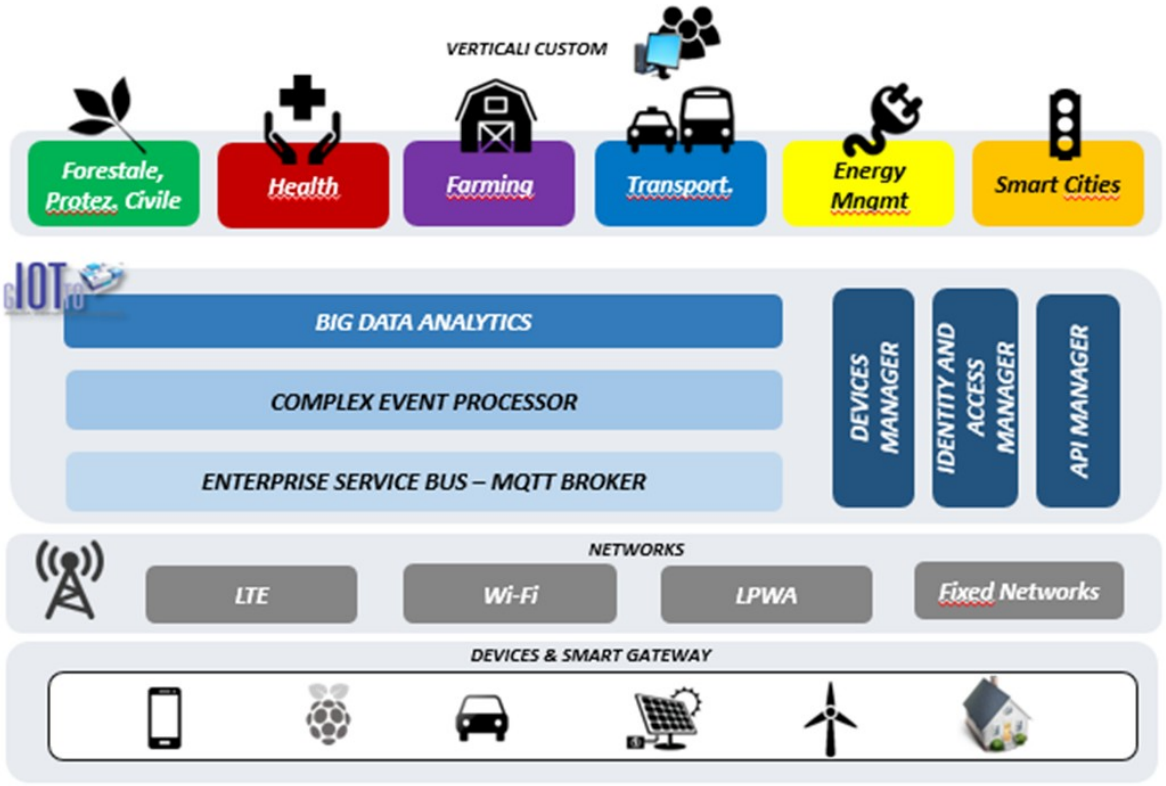
- Una descrizione dal punto di vista architetturale delle componenti della Piattaforma I.Ter IoT e I.Ter Gis fornendo il dettaglio tecnico relativo alle specifiche caratteristiche che le contraddistinguono e la valorizzano
- Le procedure di installazione e configurazione degli Ambienti necessari alla corretta messa in esercizio dei middleware facenti parti dell'Infrastruttura i.TER dislocata presso Regione Campania e adottata in Riuso dalle Amministrazioni facenti parte del partenariato I.N.T.E.R.P.A.

8.1 PIATTAFORMA I.TER IOT

I.Ter IoT è una piattaforma basata su componenti Open Source, fruibile in modalità as-a-Service, che abilita la realizzazione di una qualsiasi applicazione IoT partendo da un sistema di backend già disponibile e integrato in tutte le sue componenti. Le soluzioni e i servizi sviluppati mediante I.Ter IoT possono collezionare ed elaborare i dati generati dai sensori e dai dispositivi connessi alla rete e consentirne la fruizione da parte di utenti finali, applicazioni esterne o ancora altri device.

8.1.1 Architettura Logica

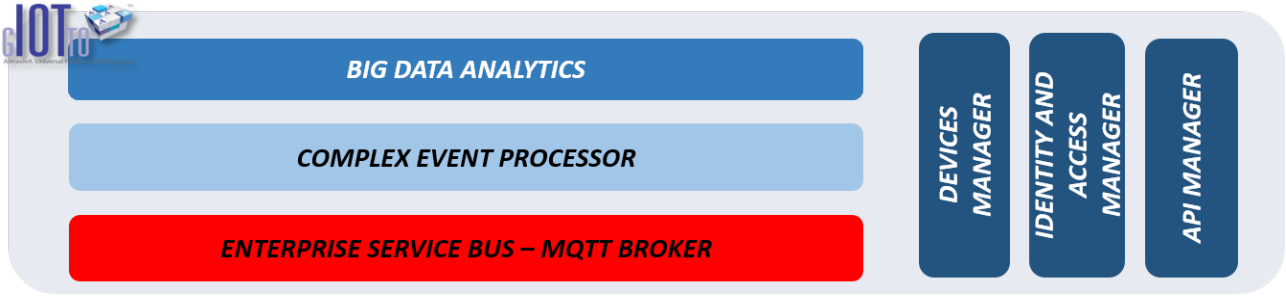
Il valore della piattaforma I.Ter IoT risiede nella composizione, in quanto è costituita da un insieme di componenti software universali e modulari che possono essere assemblati secondo le caratteristiche dello scenario da implementare; tutti gli elementi possono essere utilizzati in diverse configurazioni e per la costruzione di diverse soluzioni.



Principalmente, l'architettura è divisa in quattro layer: transport & integration, data normalization, big data & analytics, support services. Ogni layer può essere installato in maniera autonoma e può essere facilmente adattato allo scenario che si ha davanti. Ciò è reso possibile dalla scalabilità sia orizzontale che verticale di ciascun layer, una caratteristica di fondamentale importanza in un ambiente come quello dell'IoT dove ogni scenario presenta caratteristiche e requisiti diversi dagli altri.

Passiamo ora alla descrizione più di dettaglio dei diversi layer della Piattaforma I.Ter IoT.

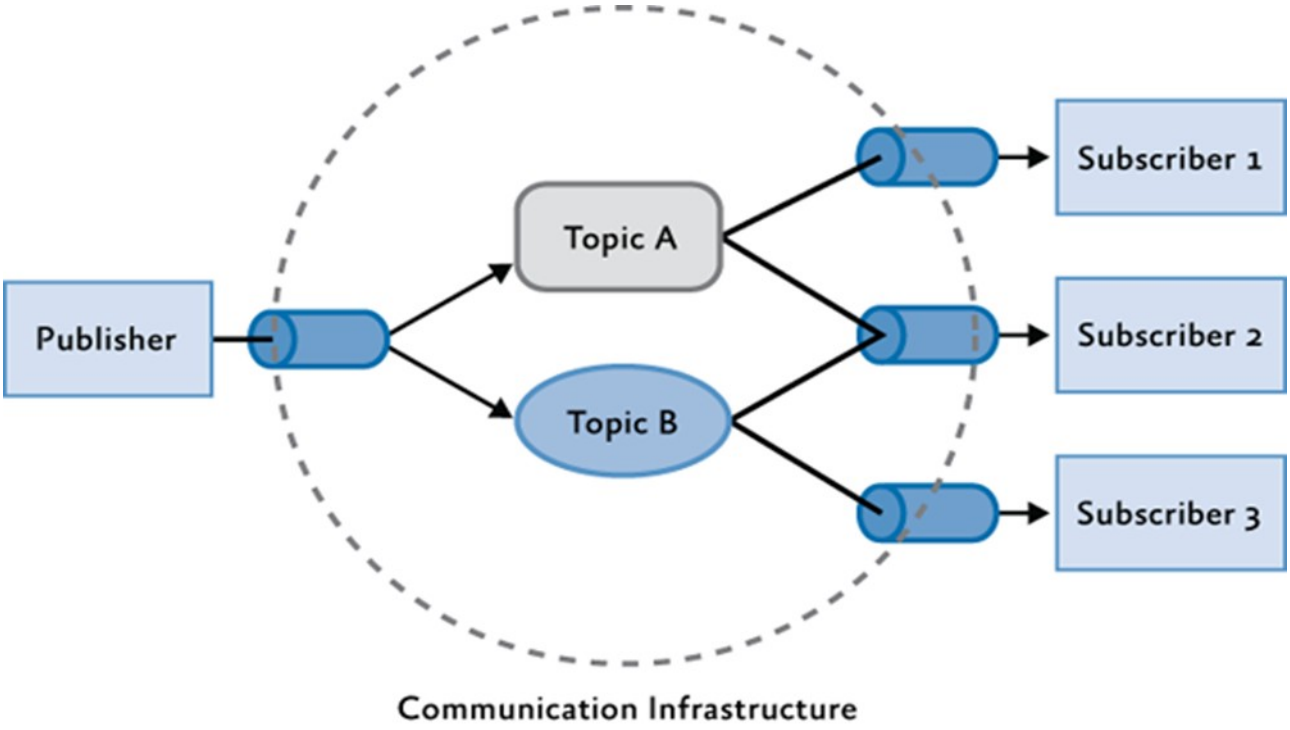
8.1.1.1 Transport & Integration



8.1.1.1.1 Message Broker

8.1.1.1.1.1 Introduzione

I.Ter IoT Message Broker è un software di messaggistica enterprise che abilita la comunicazione tra applicazioni diverse tra di loro, permettendo così la realizzazione di pattern di integrazione tra prodotti. Infatti, la messaggistica è l'aspetto più importante da tenere in conto quando si vuole garantire interoperabilità tra prodotti, per cui è necessario avere un forte strato di integrazione all'interno della piattaforma. In accordo con le più recenti tecnologie, il message broker permette alle applicazioni che lo utilizzano di comunicare in maniera asincrona e disaccoppiata.



8.1.1.1.1.2 Caratteristiche

Per quanto riguarda i protocolli supportati, I.Ter IoT Message Broker supporta i seguenti protocolli: MQTT, AMQP, Stomp, OpenWire, Web Socket, REST. Ciò permette di raggiungere un'alta flessibilità, anche perché è possibile passare da un protocollo all'altro con grande semplicità, dato che il broker al suo interno lavora con lo standard JMS 1.1. L'adozione di JMS permette di estendere in maniera universale i concetti di durabilità del messaggio, consegna del messaggio una e una sola volta, e molti altri aspetti.

Per quanto riguarda le prestazioni, I.Ter IoT Message Broker può essere opportunamente configurato per garantire diversi livelli di HA, scalabilità, affidabilità e sicurezza. Più istanze possono lavorare tra loro e, tramite l'adozione di differenti topologie di rete, adattarsi ai requisiti richiesti dallo scenario. È possibile modificare il tipo di database usato per la persistenza dei messaggi, il numero di nodi sui quali questi vengono replicati e configurare protocolli di failover per garantire un'alta disponibilità del servizio.

8.1.1.1.1.3 Vantaggi

Andiamo ad esaminare meglio quali sono i vantaggi nell'adottare I.Ter IoT Message Broker:

- Possibilità di poter scambiare messaggi utilizzando una sola componente architetturale, grazie al forte disaccoppiamento tra applicazione e modalità di messaggistica.
- API lato client disponibili in una gran varietà di linguaggi (C/C++, Java, .NET, Perl, PHP, Python, Ruby e altri ancora).
- Facilità di espansione delle feature del broker. Ad esempio, se si vuole aumentare il livello di sicurezza, è sufficiente scrivere un plugin che si pone come filtro durante la fase di autenticazione / autorizzazione.
- Possibilità di lavorare sia in modo sincrono che asincrono utilizzando una vasta gamma di protocolli. La seconda modalità è preferibile in quanto si sposa meglio con gli scenari IoT.

8.1.1.1.1.4 Modalità d'utilizzo

Nell'ambito IoT, il protocollo più utilizzato è sicuramente l'MQTT (Message Queuing Telemetry Transport), pensato per situazioni nelle quali è richiesto un basso impatto computazionale e dove la banda è limitata. Utilizza un pattern publish / subscribe appoggiandosi al message broker, ed è proprio il broker ad essere responsabile di distribuire i messaggi ai vari client. Il funzionamento di questo pattern può essere riassunto nelle seguenti fasi:

1. Il client A si connette al broker ed effettua una sottoscrizione al topic "test". Ciò equivale a dichiarare di voler ricevere i messaggi che altri client pubblicheranno su questo topic.
2. Il broker registra che il client A è interessato ai messaggi sul topic "test", per cui appena riceverà un messaggio su tale topic, provvederà a spedirlo ad A.
3. Il client B si connette al broker ed effettua una pubblicazione sul topic "test".
4. Il broker riceve sul topic test un messaggio da B. Dato che il client A è registrato sul topic "test", il broker invia ad A il messaggio che B ha pubblicato.
5. Il client A riceve il messaggio che B ha inviato.

Nell'esempio sopra descritto è evidente come il meccanismo di scambio dei messaggi avvenga in maniera asincrona. I due client non comunicano l'uno con l'altro, ma utilizzano il broker come mediatore. I client concordano con il broker i requisiti di qualità della comunicazione, e il broker garantisce che i parametri fissati vengano rispettati. Il protocollo MQTT ha diverse opzioni che gli

permettono di adattarsi a una gran varietà di scenari: ad esempio, è possibile utilizzare delle sessioni durabili per far sì che il broker si ricordi chi ha sottoscritto determinati topic ed inoltri ai sottoscrittori i messaggi che vengono ricevuti anche quando questi non sono connessi al momento di arrivo dei messaggi.

8.1.1.1.2 Enterprise Service Bus

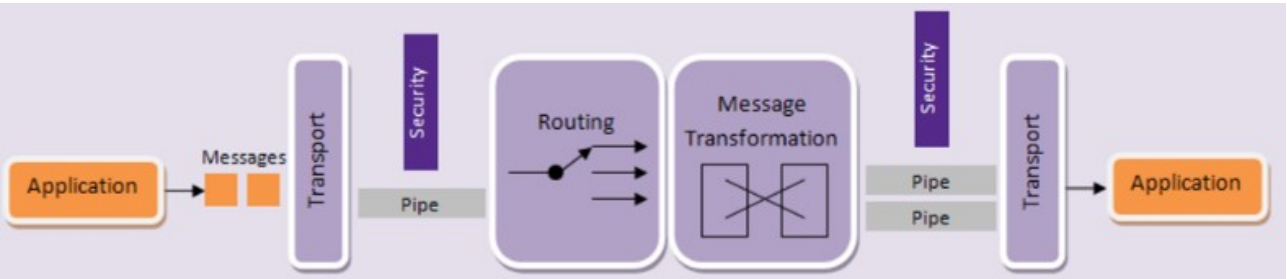
8.1.1.1.2.1 Introduzione

I.Ter IoT Enteprise Service Bus è la componente software che abilita la comunicazione tra la piattaforma e i device / smart gateway attraverso una moltitudine di protocolli e formati di messaggi affiancando il ruolo del Message Broker. Oltre a ricoprire un aspetto fondamentale come l'abilitazione al trasporto con svariate tipologie di protocolli, l'Enterprise Service Bus permette di integrare applicazioni eterogenee evitando la connessione diretta tra le stesse e scongiurando potenziali modifiche che potrebbero arrecare impatti anche consistenti sulle caratteristiche delle stesse.

8.1.1.1.2.2 Caratteristiche

Le caratteristiche principali di I.Ter IoT Enteprise Service Bus sono la versatilità, la velocità e la flessibilità; adottando i principi degli Enterprise Integration Patterns permette di realizzare una grande quantità di scenari di integrazione fra componenti.

I.Ter IoT Enteprise Service Bus adotta e supporta molteplici tipologie di protocolli di comunicazione come tcp, http, https, jms, ftp e altri ancora, abilita a scenari orientati a servizi come SOAP 1.1, SOAP 1.2, WSDL 1.1, WSDL 1.2, WS-* e REST, supporta una gestione evoluta dei messaggi di diverse tipologie e formati XML, JSON, plain text, HTML, EDI, HL7, OAGIS, Hessian, JPEG, MP4, Binary, MTOM, SwA, CORBA/IIOP, SNMP, OPC UA e altri ancora tramite filtraggio, trasformazione e routing ed è capace di interfacciarsi con i componenti più eterogenei come web service, DBMS, broker di messaggi e altri ancora.



I.Ter IoT Enteprise Service Bus, tramite appositi connettori, abilita la comunicazione verso applicazioni SaaS esterne come Twitter, Facebook, SAP, Twilio e molte altre.

8.1.1.1.2.3 Vantaggi

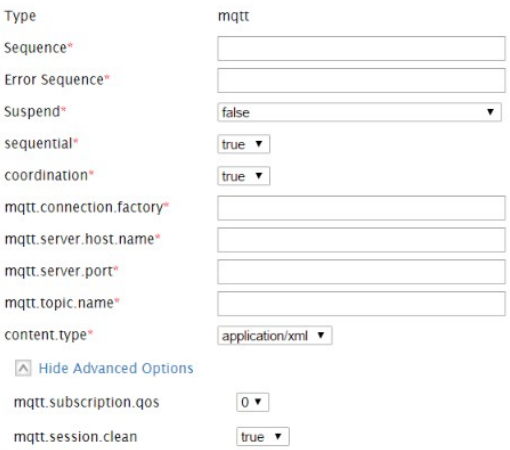
I.Ter IoT Enteprise Service Bus presenta una notevole serie di vantaggi:

- È altamente interoperabile: poiché abilita la comunicazione verso componenti eterogenee.
- È altamente performante e orientato alla scalabilità: ottimizza il consumo di risorse, supporta centinaia di connessioni contemporanee gestendole eventualmente in throttling, è ottimizzato per gestire messaggi di grande dimensione combinando tecniche di I/O non bloccante e streaming XML.
- Fornisce un supporto completo a scenari orientati ai servizi: da SOAP a REST implementa tutta una serie di standard come WS-* e una gestione evoluta dei messaggi XML e JSON; tramite XPath, JSON-Path, XSLT, XQuery permette di processare e trasformare i messaggi in ingresso e in uscita dai servizi.
- Supporta una moltitudine di protocolli, anche industriali: http/s non bloccanti, JSM transazionale, S/FTP, Mail (SMTP, IMAP, POP3), AMQP, SNMP, OPC UA, Financial Information eXchange (FIX), Hessian binary protocol per il supporto ai messaggi in formato binario.
- Offre un insieme ampio di capabilities predefinite con la possibilità di essere facilmente estese: è possibile effettuare una serie di operazioni in maniera predefinita come il routing basato sul contenuto del messaggio, il cambio di protocollo, la trasformazione del messaggio e in generale l'applicazione degli Enterprise Application Patterns. Nel caso in cui ci fosse la necessità di realizzare degli scenari specifici e non coperti dal set di funzionalità predefinite, è possibile estendere il comportamento del software (sviluppando codice ad hoc) al fine di realizzarli.

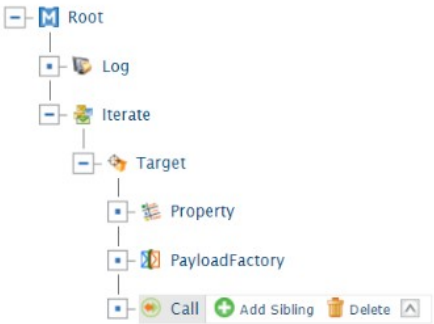
8.1.1.1.2.4 Modalità d'utilizzo

Le modalità di utilizzo di I.Ter IoT Enteprise Service Bus sono molteplici e più o meno complesse; mostriamo come questo componente software si può calare in maniera efficiente in un tipico scenario IoT.

Configuriamo l'Enteprise Service Bus per accogliere i messaggi inviati dai sensori al Broker MQTT e per farlo creiamo un endpoint di tipo MQTT specificando indirizzo, porta, credenziali di autorizzazione, livello di QoS, flag di clean session, il nome del topic, la tipologia di messaggio in input ed eventualmente le specifiche del protocollo ssl:



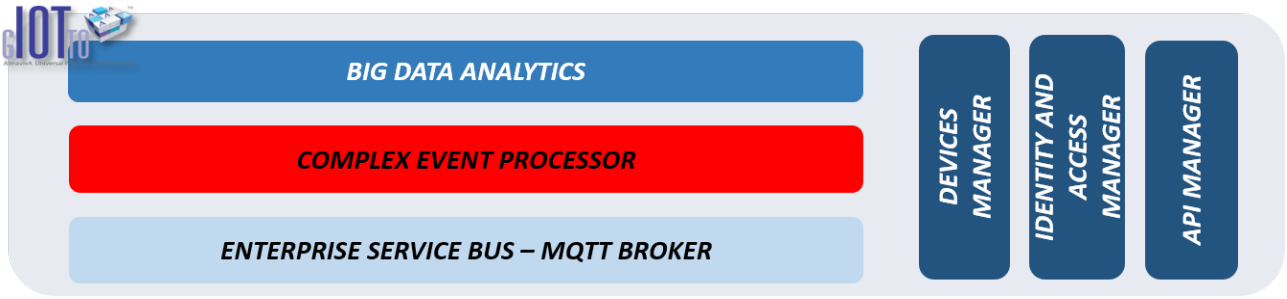
In figura è rappresentata una porzione dei parametri che è possibile specificare nella configurazione dell'endpoint MQTT. Dopo la creazione dell'endpoint è necessario creare un modello di elaborazione del messaggio in ingresso e per far ciò configuriamo una sequence, ovvero una sequenza di elaborazioni fatta sul messaggio. La sequenza di elaborazione analizza il contenuto del messaggio, lo trasforma e lo invia ad un endpoint http (potrebbe essere ad esempio un endpoint REST attraverso il quale effettuare la persistenza).



Nell'immagine in figura vediamo rappresentata la sequenza di passi per elaborare il messaggio MQTT. Per prima cosa Logghiamo il messaggio, successivamente applichiamo una regola di Iterazione del messaggio (XPath/JSON-Path) e per ogni porzione di messaggio che otteniamo andiamo a trasformarla tramite Payload Factory in un altro messaggio conforme all'input dell'endpoint da invocare con la Call. Ogni componente che abbiamo descritto presenta una maschera di configurazione, vediamo un esempio

In figura è rappresentata la maschera di configurazione del componente Iterate; nel campo Iterate Expression è possibile scrivere la regola Xpath/JSON-Path per iterare sul messaggio e avviare delle sotto sequenze con le porzioni del messaggio estratto.

8.1.1.2 Data Normalization



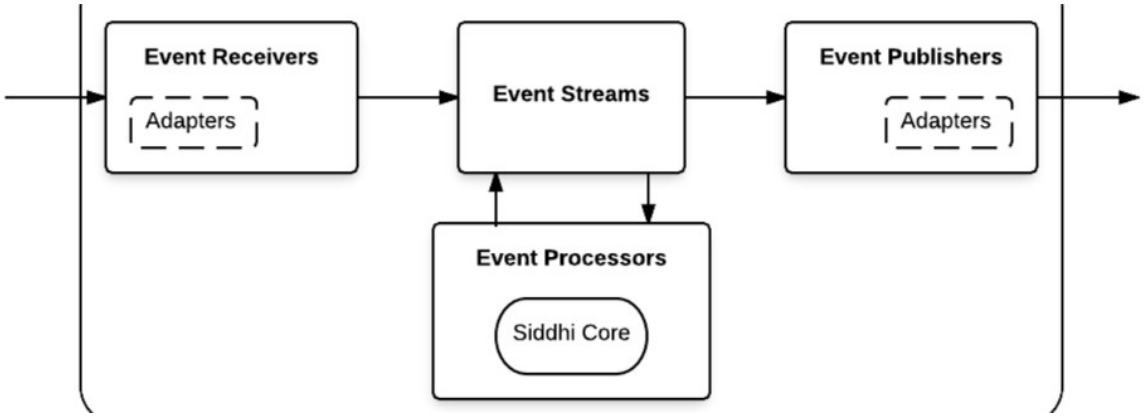
8.1.1.2.1 Complex Event Processor

8.1.1.2.1.1 Introduzione

I.Ter IoT Complex Event Processor rappresenta la componente della Piattaforma che elabora, normalizza e ritrasmette i dati ricevuti dall'Enterprise Service Bus, dal Message Broker o da componenti esterni. In base alla definizione di eventi e di regole di processamento aiuta ad identificare gli eventi più significativi e i modelli provenienti da differenti fonti di dati; analizza inoltre l'impatto degli eventi e agisce sugli stessi in tempo reale applicando meccanismi decisionali.

8.1.1.2.1.2 Caratteristiche

I.Ter IoT Complex Event Processor permette di definire Eventi, ovvero delle mappe chiave – valore che contengono i dati in ingresso alla piattaforma. Il dato in ingresso al Complex Event Processor deve essere normalizzato secondo la struttura dell'evento e questo compito è svolto dagli Event Receivers. L'Event Receiver è un modulo software che abilita alla ricezione dei dati in maniera multi protocollo; i dati possono essere accolti via http/s, mqtt, jms, web socket e altri ancora e normalizzati secondo regole XPath e JSON-Path. Sugli eventi è possibile applicare analisi e decisioni e questo compito è svolto dagli Event Processors. L'Event Processor è un motore di regole scritte in linguaggio SQL-like denominato SiddhiQL; queste regole permettono di analizzare e correlare i dati per produrre nuovi eventi, per scartare i dati non rilevanti, per prendere decisioni, per invocare determinati servizi in retroazione e altro ancora. La generazione di nuovi eventi da parte degli Event Processors scatena l'elaborazione da parte degli Event Publisher. L'Event Publisher è un modulo software speculare all'Event Receiver ma che invia i dati all'esterno del Complex Event Processor; anch'esso supporta molti protocolli tra cui http/s, mqtt, jms, rdbms, cassandra cql3 e altri ancora.



8.1.1.2.1.3 Vantaggi

I.Ter IoT Complex Event Processor presenta una notevole serie di vantaggi:

- È altamente performante e scalabile: supporta il processamento dei dati in maniera distribuita
- Adotta un motore di regole in linguaggio SQL-like: attraverso questo motore di regole permette di filtrare gli eventi in base a delle condizioni logiche, permette la creazione di nuovi eventi facendo il merge di quelli esistenti, può eseguire elaborazioni basate su finestre temporali, identifica e prende decisioni basate su pattern definiti per gli eventi e riesce a correlare i dati real-time con quelli storici presenti su delle basi dati, partiziona il carico per elaborare i dati in maniera parallela.
- Permette una gestione articolata degli eventi: l'evento viene rappresentato da delle tuple di metadati, da delle tuple di attributi di correlazione e da delle tuple che compongono il payload. Ognuna delle tuple ha un tipo primitivo di rappresentazione (intero, virgola mobile, booleano…) e può essere utilizzata dal motore di regole per calcolare dati o correlarli.
- Si integra con una moltitudine di protocolli in input: REST, JMS, SOAP, Kafka, MQTT, File, WebSocket e con messaggi di diversi formati JSON, XML, Simple Text…
- Si integra con una moltitudine di protocolli in output: REST, JMS, SOAP, Kafka, MQTT, File, WebSocket, RDBMS, NoSQL, Email, SMS e con messaggi di diversi formati JSON, XML, Simple Text…
- È possibile estenderlo con componenti custom: nel caso in cui ci fosse la necessità di realizzare degli scenari specifici e non coperti dal set di funzionalità predefinite, è possibile estendere il comportamento del software (sviluppando codice ad hoc); nello specifico è possibile creare sia connettori input/output con protocolli particolari, sia estendere il comportamento del motore di regole aggiungendo regole personalizzate.

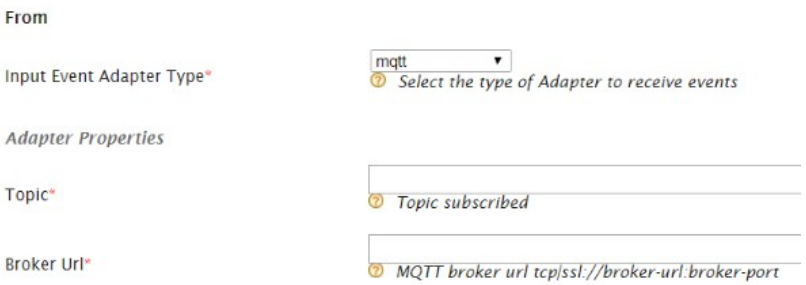
8.1.1.2.1.4 Modalità d'utilizzo

Vediamo ora come il I.Ter IoT Complex Event Processor si può calare in maniera efficiente in un tipico scenario IoT.

A tal proposito definiamo un Event Stream ovvero un insieme di tuple che conterranno i dati in ingresso alla piattaforma rappresentandoli come eventi da elaborare. Ogni proprietà ha un nome e un tipo primitivo di rappresentazione.

```
{
 "name" : "ETIG_Measure_ALL",
 "version" : "1.0.0",
 "nickName" : "",
 "description" : "Payload di test
 "payloadData" : [{
   "name" : "GTWmat",
   "type" : "STRING"
  }, {
   "name" : "GTWid",
   "type" : "STRING"
  }, {
   "name" : "GTWversion",
   "type" : "STRING"
  }, {
   "name" : "MEASid",
   "type" : "STRING"
  }, {
   "name" : "MEASints",
   "type" : "INT"
  }, {
   "name" : "MEASvalue",
   "type" : "DOUBLE"
  }, {
   "name" : "MEASstamp",
   "type" : "LONG"
  }, {
   "name" : "MEASstatus",
   "type" : "INT"
  }
 ]
}
```

Per permettere al Complex Event Processor di accogliere i dati è necessario definire un Event Receiver selezionando il protocollo di ricezione e impostando i dati necessari a stabilire la comunicazione.



Oltre ai dettagli della connessione è necessario specificare l'Event Stream che accoglierà i dati, ovvero quali eventi si genereranno all'interno del Complex Event Processor nel momento in cui arriva un messaggio all'Event Receiver.

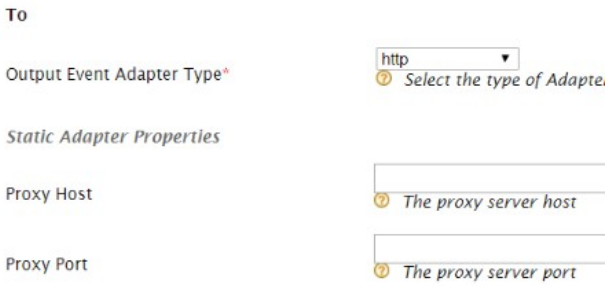
Una volta definiti Event Receiver e Event Stream è possibile procedere con la creazione delle regole di analisi degli eventi. Questo può essere fatto attraverso il linguaggio SiddhiQL nei Message Processors.

```
from inStream#window.cron('0 0/15 * 1/1 * ? *')
select GTWmat, GTWid, GTWversion, MEASid, MEASints, sum(MEASvalue) as MEASvalue, time:timestampInMilliseconds() as MEASststamp_TS, MEASstatus
group by GTWmat, MEASid
insert current events into groupedStream;
```

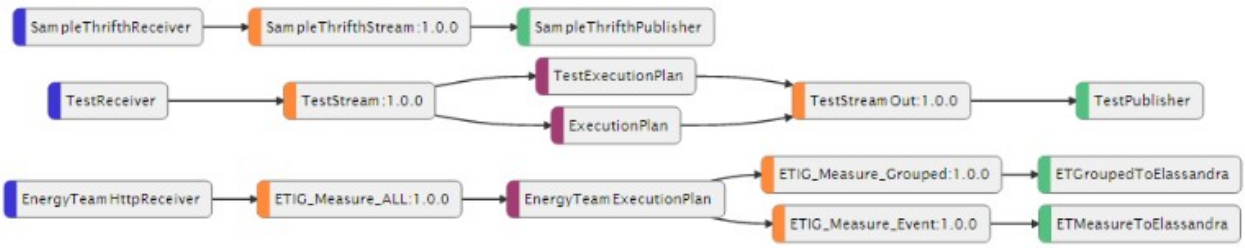
In figura è rappresentata una regola (applicata agli eventi in ingresso ad uno specifico Receiver) che definisce una finestra temporizzata secondo un'espressione CRON e sugli eventi raggruppati secondo questa finestra ne trasforma il contenuto e genera dei nuovi eventi in uscita.

Infine, sugli eventi generati dal motore di regole è in ascolto un Event Publisher il quale invierà i dati prodotti ad un output configurato verso l'esterno.

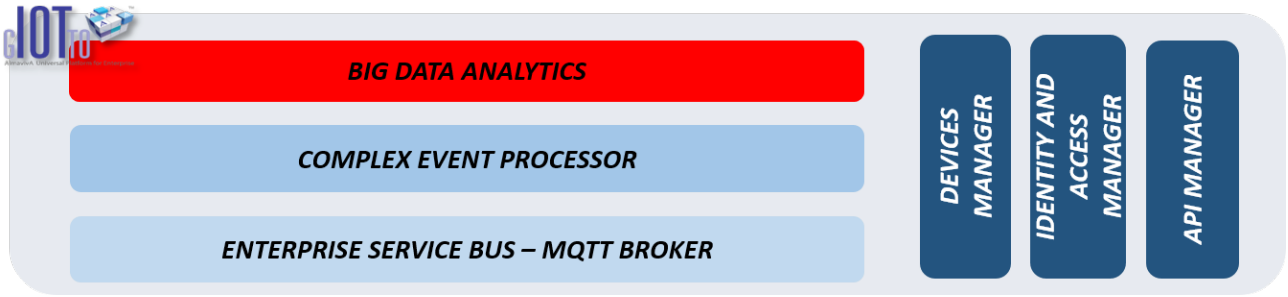
In questo caso la sorgente dei dati è l'Event Stream che tipicamente è alimentato dalle regole dell'Event Processor mentre la destinazione è un endpoint esterno al Complex Event Processor.



In qualsiasi momento è possibile osservare lo stato delle regole configurate sul I.Ter IoT Complex Event Processor. Il plugin di visualizzazione mostra lo stato dei flussi configurati dagli Event Receiver (blu) fino agli Event Publisher (verde).



8.1.1.3 Big Data & Analytics



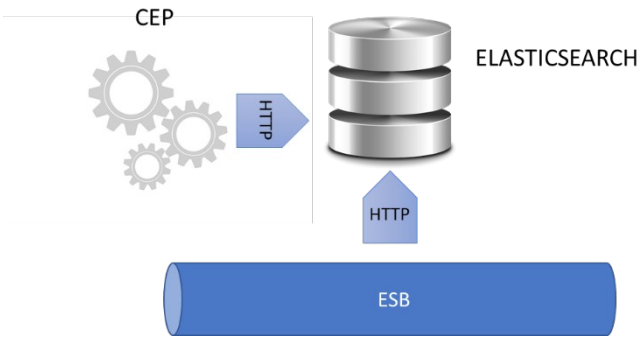
8.1.1.3.1 Data Lake

8.1.1.3.1.1 Introduzione

I.Ter IoT Data Lake rappresenta il contenitore dove raccogliere tutti i dati in ingresso alla piattaforma; questi dati possono essere successivamente sfruttati tramite algoritmi di Analytics e Machine Learning per estrarre informazioni di valore. La tipologia di Data Lake può essere scelta in base allo scenario che si vuole realizzare e ai livelli di performance in scrittura e lettura che si vogliono raggiungere.

8.1.1.3.1.2 Caratteristiche

I.Ter IoT Data Lake presenta un database di tipo nosql Elasticsearch.



Attraverso il protocollo nativo http è possibile effettuare la persistenza dei dati tipicamente sfruttando i connettori messi a disposizione dal I.Ter IoT Enterprise Service Bus e dal I.Ter IoT Complex Event Processror. I dati che arrivano alla piattaforma non devono subire alcun tipo di normalizzazione ma possono essere immagazzinati ed elaborati successivamente; le risorse computazionali per la gestione

dei dati si concentrano quindi nelle fasi successive, ovvero quando si analizzano per estrapolare informazioni di valore.

8.1.1.3.1.3 Vantaggi

Elasticsearch è un database documentale non relazionale distribuito; i vantaggi che offre sono:

- abilita alla ricerca full text all'interno dei documenti che persiste
- memorizza i documenti in formato JSON: persiste direttamente il documento che gli viene passato come parametro senza la necessità di conoscerne la struttura a priori
- espone un'interfaccia REST per la gestione dei documenti
- organizza i dati in shard ognuno con la possibilità di definire le proprie caratteristiche di replica del dato
- È resiliente: in quanto ogni fault sui nodi viene identificato e i dati replicati sui nodi attivi per mantenere il livello di replica adeguato

8.1.1.3.1.4 Modalità d'utilizzo

I.Ter IoT Data Lake può essere utilizzato come persistenza dei dati sfruttando quindi i connettori presenti nelle altre componenti della Piattaforma oppure come fonte dati per alimentare gli algoritmi di Analytics e di Machine Learning. Vediamo ora come si integrano i vari componenti della Piattaforma con il Data Lake per effettuare la persistenza dei dati.

I.Ter IoT Enterprise Service Bus s'interfaccia nativamente con le API REST di Elasticsearch tramite protocollo http. Per farlo è necessario andare ad inserire nella sequence un modulo Call attraverso il quale configuriamo l'Endpoint da chiamare, l'URI in formato template nel caso fosse necessario parametrizzarne una parte e il verbo http da utilizzare.

Per quanto riguarda l'utilizzo di Cassandra DB e HDFS è necessario estendere il comportamento del software tramite customizzazioni rappresentate dai moduli Class; questi moduli rappresentano una classe codificata che, referenziando le api dei database, gestisce il messaggio in input e ne effettua la persistenza. Nel caso in cui il messaggio in ingresso necessiti di trasformazioni prima di essere salvato è possibile utilizzare un modulo che effettua la trasformazione e poi procedere al salvataggio.

I.Ter IoT Complex Event Processor s'interfaccia nativamente con Elasticsearch tramite gli Event Publisher predefiniti ma necessita di estensioni ad hoc per connettersi ad HDFS e a Cassandra DB tramite protocollo CQL3.

8.1.1.3.2 Analytics

8.1.1.3.2.1 Introduzione

I.Ter IoT Analytics è la componente della Piattaforma finalizzata alla ricerca, all'interpretazione e all'estrapolazione di pattern significativi all'interno dei dati persistiti nel Data Lake. I.Ter IoT Analytics garantisce un'elevata efficienza nell'esecuzione di algoritmi iterativi e sull'analisi interattiva dei dati.

8.1.1.3.2.2 Caratteristiche

I.Ter IoT Analytics presenta un'interfaccia user-friendly verso variegate tecnologie di backend come Hadoop, Spark, Elasticsearch, R Studio e altre ancora; attraverso wizard grafici permette di creare query in maniera semplice e intuitiva che abilitano ad un rapido sviluppo di verticali attraverso i quali prospettare i risultati in formato tabellare e/o grafico. Attraverso la componente R Studio permette di applicare algoritmi standard come l'Anomaly Detection, il Forecast e permette anche di implementare algoritmi custom da calare nel contesto di riferimento.

I.Ter IoT Analytics presenta una web application per la creazione della reportistica sui dati

denominata Redash.

8.1.1.3.2.3 Vantaggi

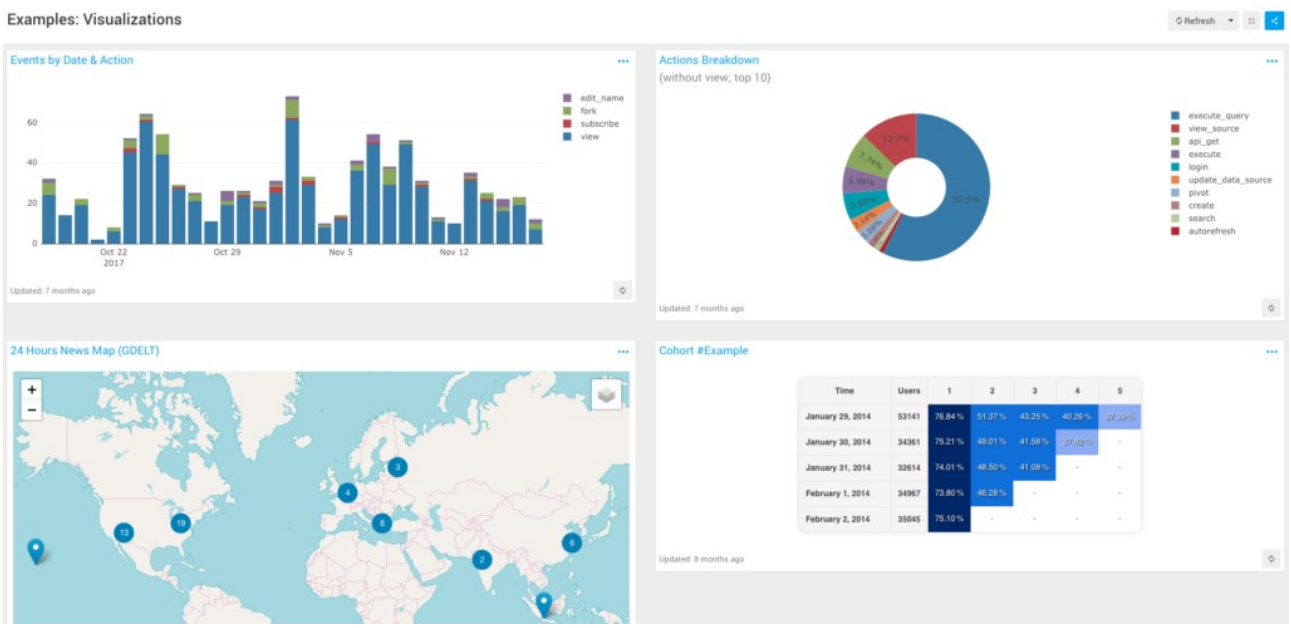
Elenchiamo di seguito alcuni vantaggi che offre I.Ter IoT Analytics:

- È altamente scalabile: garantisce ottime performance all'aumentare della quantità dei dati
- Abilita alla ricerca full-text su dati strutturati e destrutturati: ad esempio su file PDF
- Permette di collegare dataset differenti con un click: ovvero permette di effettuare delle JOIN virtuali
- Suggerisce eventuali correlazioni tra i campi dei dataset: per fare ciò utilizza un Data Dictionary nel quale tiene le informazioni dei dati attraverso le quali identifica le corrispondenze
- Garantisce la massima libertà nella manipolazione dei dati: non richiede che le KPI vengano definite a priori
- Agisce da repository universale: nel senso che permette di elaborare/incrociare i dati provenienti da sorgenti differenti

Per quanto riguarda la reportistica, Redash è una web application di data visualization e dashboarding che consente di scrivere le query nella loro sintassi naturale per interrogare le basi dati. L'applicativo è ricco di funzionalità come auto-completamento in tempo reale, e molte scorciatoie da tastiera, consente inoltre di creare snippet per elementi usati di frequente così da poterli riutilizzare.

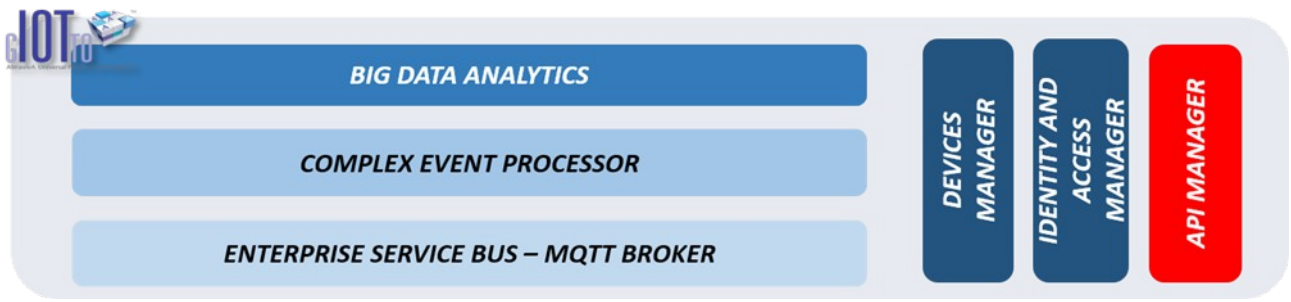
8.1.1.3.2.4 Modalità d'utilizzo

Con I.Ter IoT Analytics è possibile visualizzare i dati secondo diverse modalità come, ad esempio: visualizzazione in modalità Data Discovery, visualizzazione per PKI, etc.



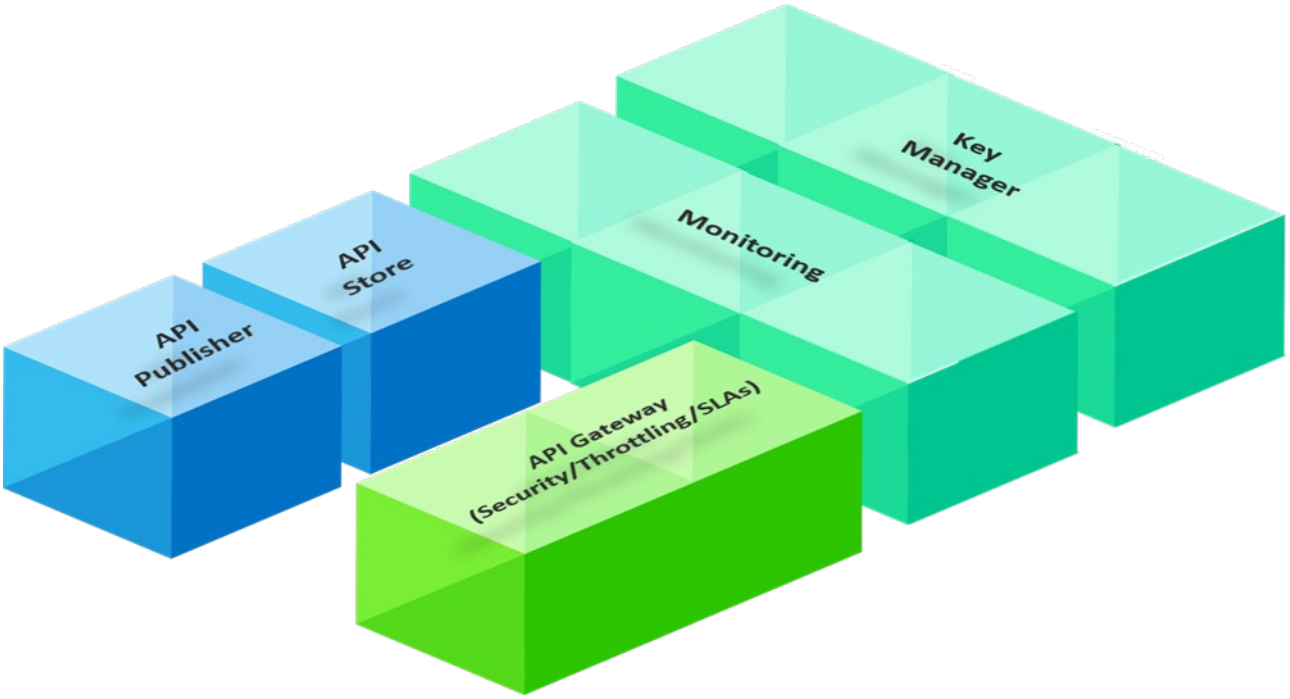
8.1.1.4 Support Services

8.1.1.4.1 API Manager



8.1.1.4.1.1 Introduzione

I.Ter IoT API Manager è una soluzione che permette di creare, pubblicare e gestire tutti gli aspetti delle API e del loro ciclo di vita, permettendo anche di poter scalare in caso di scenari massivi. Quando si lavora con applicazioni grandi e complesse, è assolutamente necessario esercitare il controllo, stabilire la fiducia, mettere in sicurezza e regolare le API in modo che esse possano essere utilizzate in modo collaborativo. Inoltre, dal punto di vista commerciale è necessario avere una policy di monetizzazione delle API, dato che queste sono strettamente collegate con l'uso che viene fatto di un servizio o di un'applicazione: I.Ter IoT API Manager permette sia di gestire la monetizzazione che il throttling, ossia una pratica necessaria per prevenire l'uso eccessivo delle API.



8.1.1.4.1.2 Caratteristiche

Per soddisfare il compito, I.Ter IoT API Manager utilizza più componenti: API publisher, API store, API gateway, Key Manager, Traffic Manager. Lo sviluppo delle API è solitamente eseguito da qualcuno che capisce gli aspetti tecnici delle API, mentre la gestione delle API è effettuata da qualcuno che capisce l'impatto economico.

La componente di API publisher include sia la parte di sviluppo che la parte di gestione delle API. Permette di gestire lo sviluppo, la documentazione, lo scaling e il versionamento delle API, facilitando allo stesso tempo i compiti di gestione come la pubblicazione, la monetizzazione e l'analisi delle statistiche dell'utilizzo.

La componente API store si pone come step successivo alla componente precedente. Una volta che un set di API è stato pubblicato, gli utenti possono visualizzarlo, vederne la documentazione, sottoscriverlo e successivamente valutare le API che ne fanno parte.

Per gestire le chiamate agli endpoint, si utilizza la componente di API gateway. Essa ha il compito di proteggere, gestire e mettere in sicurezza le chiamate. Intercetta le richieste e applica policies di throttling e sicurezza, utilizzando degli handler diversi a seconda della situazione. Se il gateway fa passare la chiamata, questa viene inoltrata direttamente al backend, a meno che la chiamata non

riguardi la richiesta di un token.

In questo ultimo caso, interviene il key manager, che ha il ruolo di gestire la sicurezza, il rilascio e la gestione dei token. Dopo aver sottoscritto un'API, è possibile utilizzare le chiavi per ottenere l'access token, che poi dovrà essere utilizzato ogni volta che si vorrà invocare il servizio. Il gateway lavora a stretto contatto con il key manager, dato che la verifica dell'access token viene effettuata ogni volta che un'API corrispondente all'applicazione sottoscritta viene richiamata. Tutti i token utilizzati per la validazione sono basati sullo standard OAuth2, ossia lo standard de-facto per quanto riguarda l'interazione tra applicazione e utente.

Infine, il traffic manager rende disponibili le API a livelli di servizio differenti a seconda del consumer e mette in sicurezza le API contro attacchi di sicurezza. Utilizza un engine di throttling dinamico, che gli consente di soddisfare in tempo reale i requisiti stabiliti nelle policies.

8.1.1.4.1.3 Vantaggi

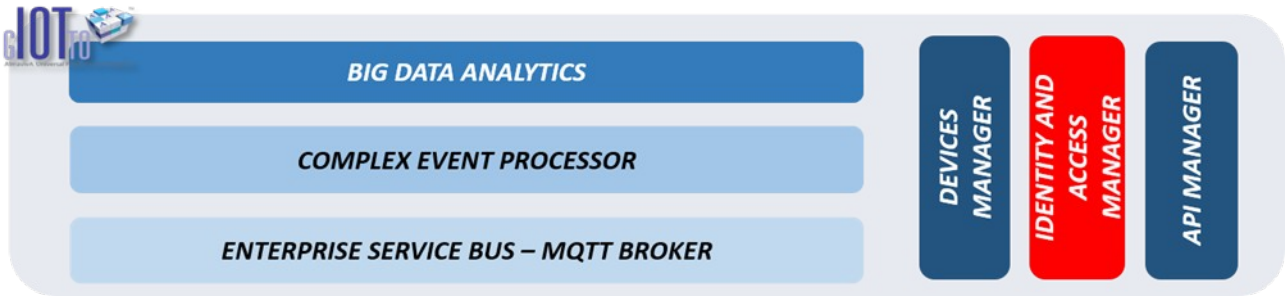
I.Ter IoT API Manager permette di introdurre numerosi vantaggi, soprattutto in un'architettura IoT:

- Minor peso sul lavoro dello sviluppatore, in quanto non deve concentrarsi su aspetti di sicurezza e di gestione del traffico.
- Facilità nel compito di monetizzazione e monitoraggio di un servizio, resa possibile dall'engine di throttling.
- Possibilità di gestire il ciclo di vita completo delle API, dallo sviluppo alla produzione fino alla user experience, il tutto utilizzando un unico prodotto.
- Scalabilità delle singole componenti in funzione dello scenario: ad esempio, si possono aumentare le componenti gateway in uno scenario dove il numero di richieste è molto elevato.
- L'utilizzo di OAuth2 introduce un modo di proteggere le API flessibile e rivoluzionario, permettendo a chi sottoscrive le API di non fornire le proprie credenziali ogni volta che deve accedere ad un servizio.

8.1.1.4.1.4 Modalità d'utilizzo

In uno scenario IoT, un esempio di utilizzo di I.Ter IoT API Manager è quello della protezione dei dati esposti da un servizio di backend. Supponiamo che dopo aver raccolto, filtrato e normalizzato dati provenienti da migliaia di sensori, questi vengano persistiti dentro Elastic Search. Un servizio esterno chiede di agganciarsi alla piattaforma I.Ter IoT, e di poter leggere i dati che sono stati acquisiti: tramite l'utilizzo di I.Ter IoT API Manager è possibile esporre solo determinati endpoint, in modo che solo gli utenti autorizzati possano leggere i dati. Inoltre, è possibile fissare un limite alle letture che verranno effettuate, in modo da creare una sorta di monetizzazione dei dati esposti, che di base non è prevista da Elastic Search. In generale, utilizzando questo approccio è molto facile integrare qualsiasi servizio esterno e renderlo parte della piattaforma mantenendo gli stessi utenti e ruoli utilizzati anche per gli altri servizi, a patto che quest'ultimo esponga degli endpoint http.

8.1.1.4.2 Identity Server



8.1.1.4.2.1 Introduzione

I.Ter IoT Identity Server è un prodotto che permette di gestire le identità utilizzate all'interno della piattaforma I.Ter IoT, il tutto in un'ottica multitenant. All'interno di ogni tenant è possibile creare utenze ed assegnare a queste uno o più ruoli: i permessi concessi a ciascun ruolo permettono di usufruire delle funzionalità offerte dalla piattaforma. È possibile agganciare più user store all'Identity Server (Apache DS, Open LDAP, Active Directory, SQL database), in modo da poter soddisfare l'integrazione con un gran numero di prodotti. È altamente integrato con il I.Ter IoT API Manager e con OAuth2, consentendo anche di scrivere plugin per gestire grant e regole di accesso personalizzate.

8.1.1.4.2.2 Caratteristiche

Per garantire piena compatibilità con tutti i tipi di gestione di utenze, I.Ter IoT Identity Server supporta tre tipi di controllo di accesso: XACML (extensible access control markup language), RBAC (role based access control) e ABAC (attribute based access control). Supporta inoltre tutti i principali flussi OAuth, fornendo tutti gli endpoint necessari per la realizzazione di tali flussi.

Una delle feature più importanti è quella dell'SSO, che permette agli utenti di fornire le credenziali solo una volta e ottenere l'accesso a molteplici applicazioni. In questo modo agli utenti vengono chieste le credenziali solo una volta, e, fino al termine della sessione, possono accedere a tutte le componenti della piattaforma se il ruolo a loro assegnato lo consente.

Qualora non si volesse utilizzare il sistema interno di autenticazione, è possibile agganciare un identity provider esterno come Facebook, Google, Yahoo, LinkedIn o altri ancora. I.Ter IoT Identity Server può fornire in questo modo l'integrazione della piattaforma I.Ter IoT con federated authenticator esterni, aumentando la decentralizzazione dell'identità qualora fosse richiesto dall'applicazione.

8.1.1.4.2.3 Vantaggi

L'inclusione di I.Ter IoT Identity Server nella piattaforma permette di ottenere, in modo semplificato e sicuro, delle specifiche di sicurezza facilmente estendibili al resto dell'architettura:

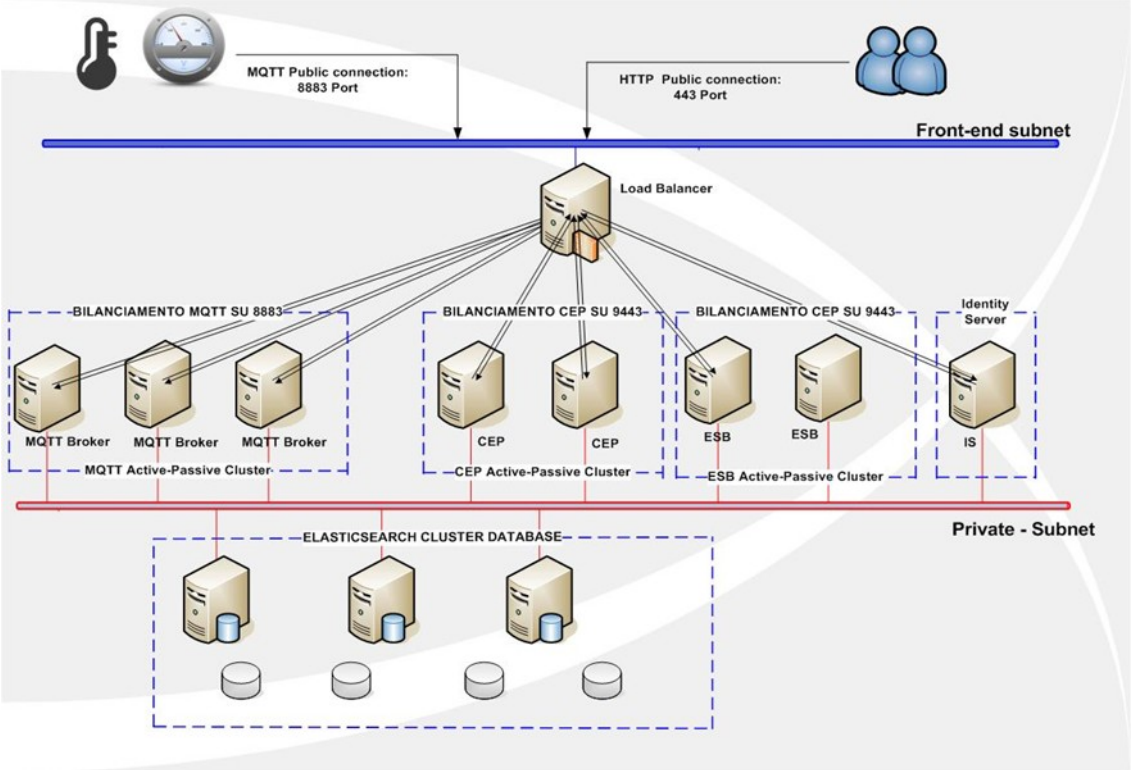
- Single Sign On e Single Log Out basato su SAML2.
- Possibilità di aumentare il livello di sicurezza utilizzando procedure di autenticazione multi step, come ad esempio certificati X.509 e TOTP (time base one time password).
- Disaccoppiamento tra user store e implementazione dell'applicazione per quanto riguarda la richiesta di permessi.

8.1.1.4.2.4 Modalità d'utilizzo

Un utilizzo semplice ma allo stesso tempo utile ed efficace di I.Ter IoT Identity Server è il suo utilizzo all'interno del protocollo MQTT per gestire l'autenticazione e l'autorizzazione al broker. Dato che tutta la piattaforma utilizza la stessa struttura per i ruoli, diventa molto facile modificare i permessi di lettura e scrittura sui topic senza dover cambiare ogni volta le impostazioni del message broker. Basta effettuare una sola volta il mapping all'interno del broker tra i suoi ruoli e i ruoli offerti dalla piattaforma; se in futuro si vuole negare il permesso di lettura di un topic, ciò può essere fatto in maniera snella e senza dover riavviare niente, modificando semplicemente il ruolo assegnato all'utente che si vuole limitare.

8.1.2 Architettura Fisica

Vediamo di seguito il disegno dell'architettura fisica del modulo IoT:



Le macchine che compongono la Piattaforma sono distribuite su una subnet privata e rese accessibili attraverso un bilanciatore attestato su una subnet di front-end. I layer di integrazione (MQTT Broker, CEP e ESB) e di big data sono configurati in alta affidabilità in maniera tale da garantire la tolleranza ai guasti e la continuità del servizio.

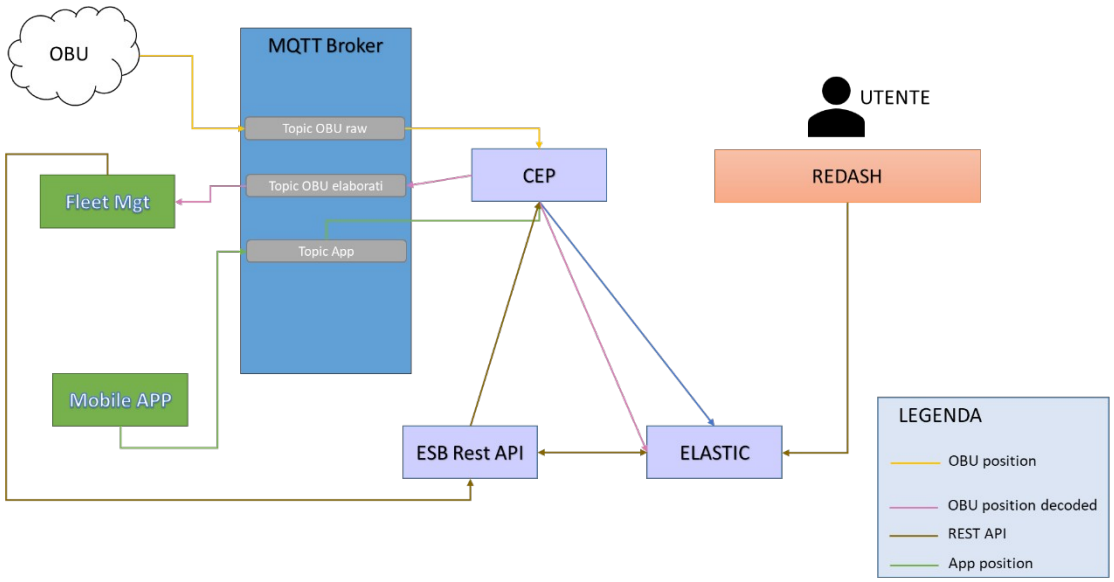
Vediamo di seguito una tabella che descrive le caratteristiche delle virtual machine che compongono la Piattaforma I.Ter IoT:

Ruolo VM	Nome VM	S.O	RAM	CPU	HD

MQTT-Broker	itermqtt1	CENT OS	8	4	40
Complex Event Processor	itercep1	CENT OS	8	4	60
Enterprise Service Bus	iteresb2	CENT OS	8	4	60
Database Elasticsearch	iterdata3	CENT OS	8	4	40
Identity Server	iteris	CENT OS	8	4	40
MySQL					
Redash					

8.1.3 Flussi di piattaforma

Di seguito una rappresentazione dei principali flussi di piattaforma



8.2 PIATTAFORMA I.TER GIS

L'aplicativo i.TER GIS è una applicazione GIS web enterprise, multiutente e multiprofilo.

Ciò significa che è i.TER GIS consente ad ogni utente (secondo i privilegi assegnatigli) la possibilità di gestire i propri Progetti, intesi come collezione di layers cartografici organizzabili logicamente mediante Nested Folders.

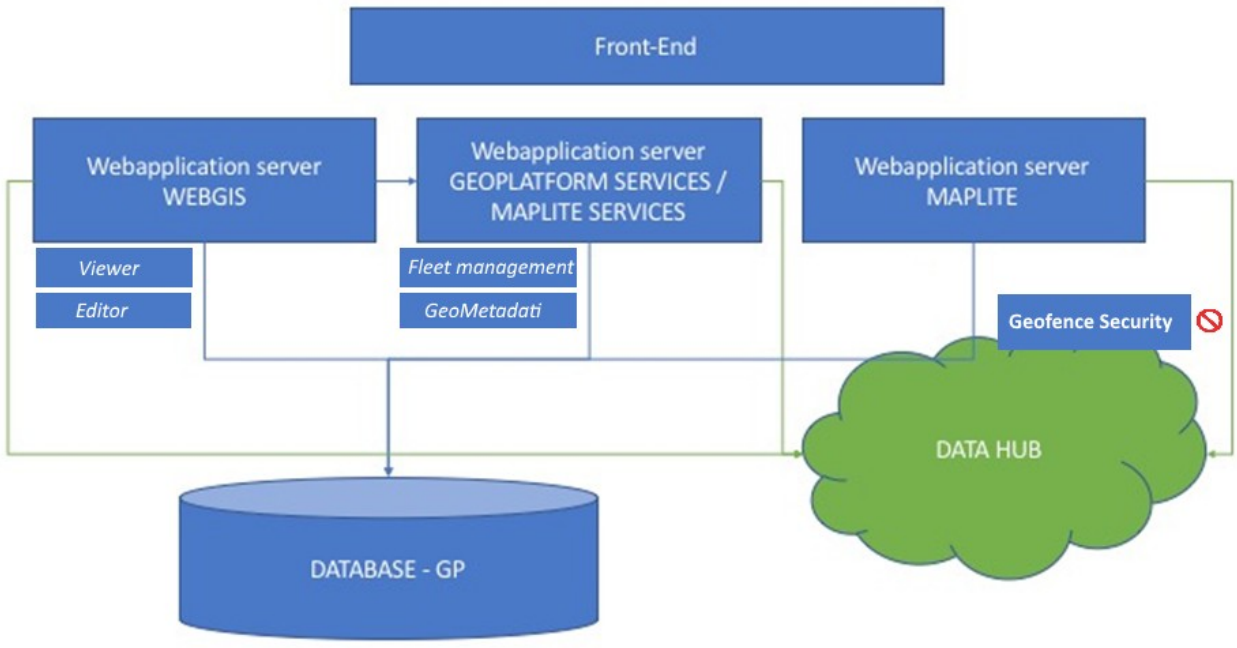
Mediante la funzione Gestione Progetti, l'utente può creare progetti (sia da zero che a partire da progetti pre-esistenti), impostandone le proprietà, la versione , la struttura delle Nested Folders, i layers cartografici contenuti.

la funzione, inoltre, è anche responsabile di facilities di esportazione e importazione della struttura dei progetti, mediante le quali un utente può salvare localmente un file contenente tutte le informazioni relative al progetto e ai layer contenuti, fissandone la versione e le proprietà, e disporre di tale file per un successivo import in altro progetto o per condividerne la struttura con altri utenti.

Ogni utente può gestire un numero illimitato di progetti mappa (costituiti da folders e layers). Collegandosi con le proprie credenziali ha accesso al progetto da egli indicato come default, e può in ogni momento passare alla visualizzazione di un altro progetto, mediante selezione dalla lista dei suoi progetti.

8.2.1 Architettura Logica

L'architettura prevede i macro-componenti riportati in figura:



L'accesso a i.TER GIS avviene tramite Front End, che si occupa di indirizzare le richieste al **Web Application Server (WAS GIS).** Questi si occupa di effettuare il dispatching delle richieste del frontend ad altri componenti quando sono di loro competenza (es. funzionalità svolte da macro-moduli), oppure di elaborarle direttamente nel caso di elaborazioni di visualizzazione o querying.

Il WebGIS ha interazioni con altri componenti backend:

- **Geo-Platform Services,** sono i servizi che colloquiano con l'esterno, ad esempio con i server WMS per raccogliere informazioni massive (Es. WMS GetCapabilities) e presentarle alla parte client del WEBGIS.
- **Maplite,** è un altro applicativo WebGIS che consente la pubblicazione (accesso anonimo) di progetti mappa creati tramite WebGIS.
- **GIS Viewer,** modulo integrato nel WAS GIS che sottende a tutte le operazioni di visualizzazione per quel che concerne le pertinenze della mappa sul browser (layer, stili, zoom e simili)
- **GIS Editor,** modulo integrato nel WAS GIS che sottende a tutte le operazioni inerenti l'editing collegato agli oggetti spaziali per cui è previsto.
- **Fleet Management,** collezione di tecnologie che fornisce tutte le opzioni (da quelle di base alle più avanzate) legate alla gestione del tracking su percorsi ed alle funzionalità a questo collegate
- **GeoMetadati,** per la gestione della meta datazione dei dati spaziali.
- **Data Hub,** si tratta del punto centrale di creazione ed erogazione di servizi di mapping

conformi agli standard dell'Open Geospatial Consortium (OGC®) WMS (Web Map Service), WFS (Web Feature Service), WCS (Web Coverage Service). Consente inoltre la Tematizzazione delle mappe geografiche erogate in WMS mediante l'applicazione dello standard SLD (Style Layer Description). Il dato spaziale così erogato viene generato a partire da diversi data souce, come cache da filesystem o dati spaziali immagazzinati nel db.

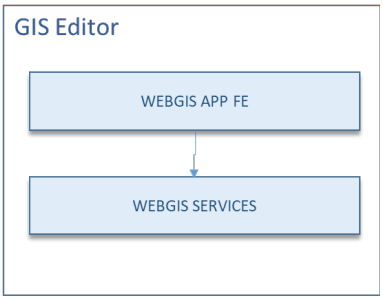
- **Geofence Security.** Componente declinato coi software Geofence per Geoserver ed Almaviva SDUF atto a garantire l'accesso al dato geografico in funzione delle credenziali del chiamante. il filtering può avvenire sia nella modalità intero layer (mostra o nasconde un layer ad un utente) oppure su un sottoinsieme di dati contenuti nel layer.
- Database **GP,** è qui che Il WebGIS ha le proprie informazioni di persistenza

Naturalmente tutte queste componenti hanno accesso ai servizi del data HUB

8.2.2 GIS VIEWER, MAPLITE E GIS EDITOR

Il core della piattaforma i.TER Campania è costituito dalle interazioni tra moduli che fanno emergere le funzionalità di visualizzazione che di editing del dato spaziale. Ciò Consente la visualizzazione su mappa, creazione, condivisione e manipolazione di dati geografici, nonché la condivisione di progetti e mappe digitali create dagli utenti della piattaforma. Grazie all'interoperabilità con servizi OGC-compliant

8.2.2.1 Il GIS Editor



Il modulo è composto dalla componente WebGIS e dalla componente geoplatform; la prima costituisce l'interfaccia attraverso cui l'utente interagisce con il sistema ed effettua le operazioni di caricamento visualizzazione e manipolazione del dato geografico; la seconda mette a disposizione delle varie componenti i servizi per accedere al layer di persistenza e al datahub. Gli strumenti a disposizione dell'utente rendono semplice la modifica degli oggetti e la loro persistenza nel GeoDB.

8.2.2.2 Il GIS Viewer ed il MAPLITE

La Componente Viewer ha due declinazioni, la prima orientata agli utilizzatori dotate di opportune

credenziali, grazie alle quali possono accedere utenti che possono trattare dati non pubblici secondo le prerogative dei propri privilegi. La seconda è estesa con un ulteriore componente detto MAPLITE. Questo strumento consente, attraverso la creazione di un url pubblico, la visualizzazione di un progetto (insieme strutturato di più strati informativi) creato all'interno del GIS editor ad un qualsiasi utente web non autenticato. L'url creato determina un collegamento diretto con il progetto, assimilando pertanto qualsiasi variazione venga effettuata dall'owner del progetto. Lo strumento possiede diverse funzionalità legate all'interrogazione e la visualizzazione delle informazioni geografiche contenute negli strati informativi che compongono il progetto nonché la possibilità di scegliere tra le principali basemap in circolazione (Bing, OSM, e comunque dotate di licenza utilizzabile nel caso d'uso) su cui proiettare gli strati informativi presenti. E' dotato poi di una serie di toolbox che agevolano la visualizzazione, l'interazione e l'integrazione dei componenti.

8.2.3 METADATI

Lo strumento di metadatazione delle informazioni geografiche. Il sistema di metadatazione è basato sulle linee guida INSPIRE e consente a chi "produce" il dato di certificarne le caratteristiche e a chi lo "consuma" una maggiore comprensione del medesimo, aiutando pertanto la ricerca e la scoperta dei dati stessi. La metadatazione favorisce inoltre l'interoperabilità con le altre banche dati della Pubblica Amministrazione.

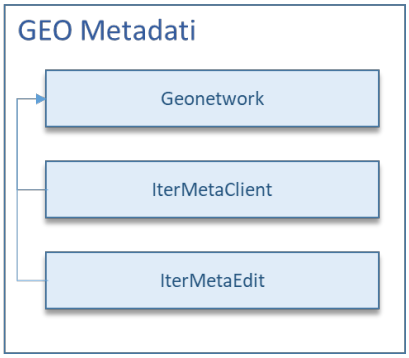
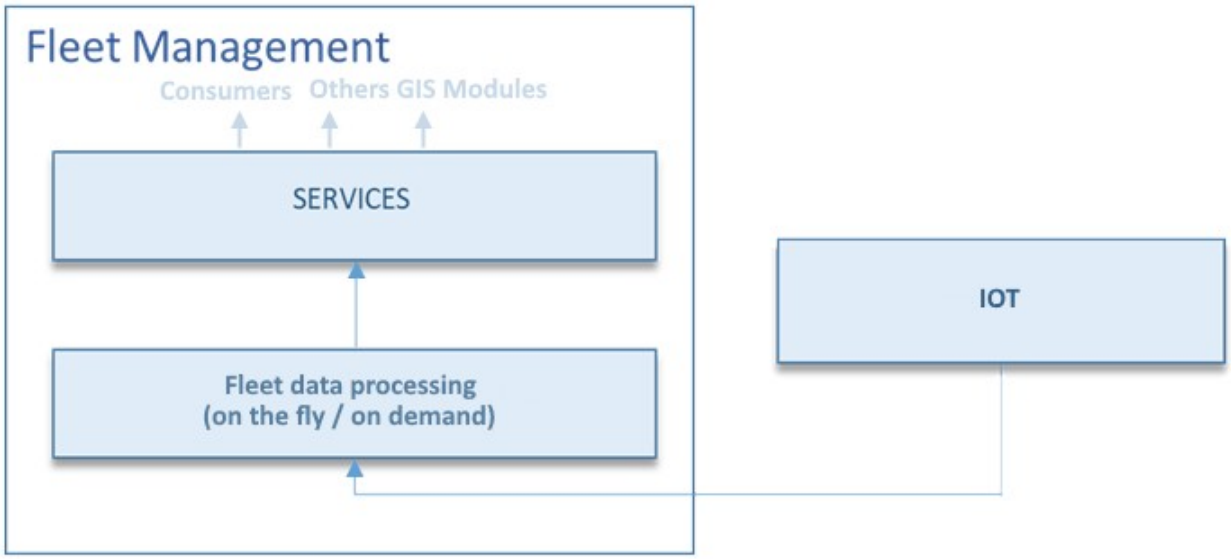


Figura 6 - GEO Metadati

Il modulo geometadati è costituito da 3 moduli: geonetwork, iterMeta_Client e IterMeta Edit. La componente geonetwork è responsabile delle operazioni di harvesting dal DataHub delle informazioni candidate a diventare metadati. La componente IterMeta_Client consente la visualizzazione di tutti i layer raccolti dal geonetwork e consente di attivare la componente IterMeta Edit per la metadatazione dei layer.

8.2.4 FLEET MANAGEMENT

L'insieme delle tecnologie e funzioni atte alla gestione del tracking di veicoli o dispositivi in movimento su percorsi predefinibili è detto Fleet Managenment. Più specificatamente il componente facente parte dell'architettura in esame supporta la visualizzazione del dato in real-time ed il suo dataming spaziale sia in modalità "on the fly" (alert percorsi non schedulati) che on demand (reportistica dei kilometri percorsi e simili).



Come da figura, il componente preleva il dato dalla componente IOT, ne applica il processing secondo le modalità prima esposte, ed espone i risultati come servizi. Da qui essi possono confluire su vari consumer: come il Viewer GIS che renderizza gli attributi latitudine e longitudine come punto su mappa assieme ai percorsi in attraversamento, o come algoritmi che attraverso querying geospaziale ed operazioni di cumulo forniscono statistiche e report circa i percorsi effettuati da un elemento monitorato durante un certo periodo temporale. Da quanto detto il Componente Fleet managment è ad uso diretto per la visualizzazione, ad uso indiretto poiché si comporta come un modulo propedeutico ad ulteriori processamenti e relative funzioni.

8.2.5 DATABASE

Il database è dotato di funzionalità GIS sia per lo storage che per l'interrogazione ed il dataming spaziale. Esso è inoltre logicamente diviso in 3 componenti fondamentali:

- Metadata applicativo: contiene tutte le informazioni proprietarie del sistema (strutture

dati, progetti, reportistiche, etc…).

- Geodatabase: contiene tutti dati cartografici necessari alla rappresentazione geografica dei dati.
- Data warehouse: contiene i dati su cui il sistema permette di effettuare databrowsing e produzione indicatori.

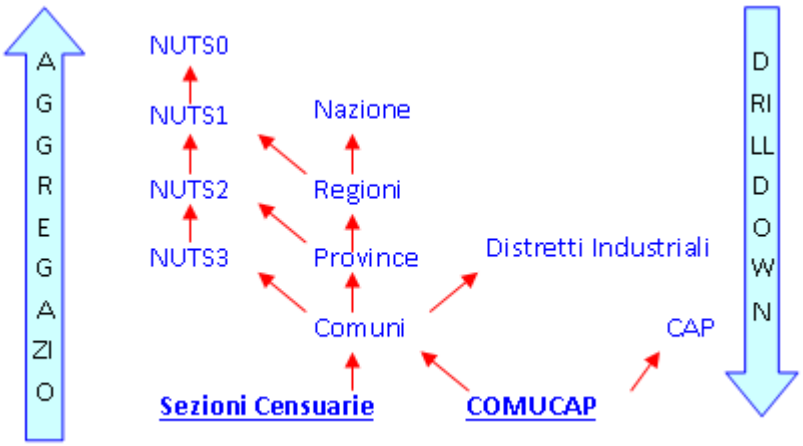
Il modello dati è il "modello snowflake" di database per il databrowsing. Questo modello prevede una o più tabelle degli oggetti, ognuna con dimensioni e variabili. Ogni dimensione è una chiave esterna ad una tabella che codifica i diversi valori che si possono trovare nella dimensione. In genere queste tabelle vengono chiamate livelli di dettaglio o nodi di gerarchia. Le tabelle (nodi di gerarchia) possono essere collegate tramite relazioni ad altre tabelle padri (altri nodi della gerarchia). In questo modello non sono permesse le relazioni circolari.

8.2.5.1 Dati Cartografici

Ogni livello di dettaglio territoriale (che riferisce elementi del territorio) può avere una o più cartografie associate, di tipo puntuale, multipoint, lineare e poligonale. Ad esempio il livello Comuni (NOD_COMUNI) può avere associato la cartografia comunale poligonale su scala 1:1.000.000 e su scala 1:250.000, oppure, il livello stazioni di monitoraggio può avere una cartografia puntuale e una cartografia poligonale (con l'area di pertinenza).

Ogni livello territoriale deve avere associato un dato cartografico se si vogliono fare delle tematizzazioni cartografiche su quel livello. I dati cartografici possono avere un riferimento di validità temporale. In questo modo è possibile creare cartografie che evolvono nel tempo, e quindi possono rappresentare correttamente dati che hanno una dimensione temporale.

La cartografia delle unità territoriali di riferimento permettono di rappresentare l'informazione analitica (indicatori e altri dati) su una base territoriale in forma di tematizzazioni, grafici e tabulati. Di solito vengo scelti come unità territoriali di riferimento dati vettoriali poligonali che hanno un interesse particolare dal punto di vista amministrativo, socio-economico o strutturale. Queste unità territoriali possono essere in relazione di composizione (aggregazione) fra di loro. Ciò permette l'implementazione di meccanismi di aggregazione di dati analitici e di stima di indicatori in base alla distribuzione territoriale dei dati. Fra queste relazioni si possono individuare gerarchie che definiscono modalità in cui si aggregano i dati analitici.

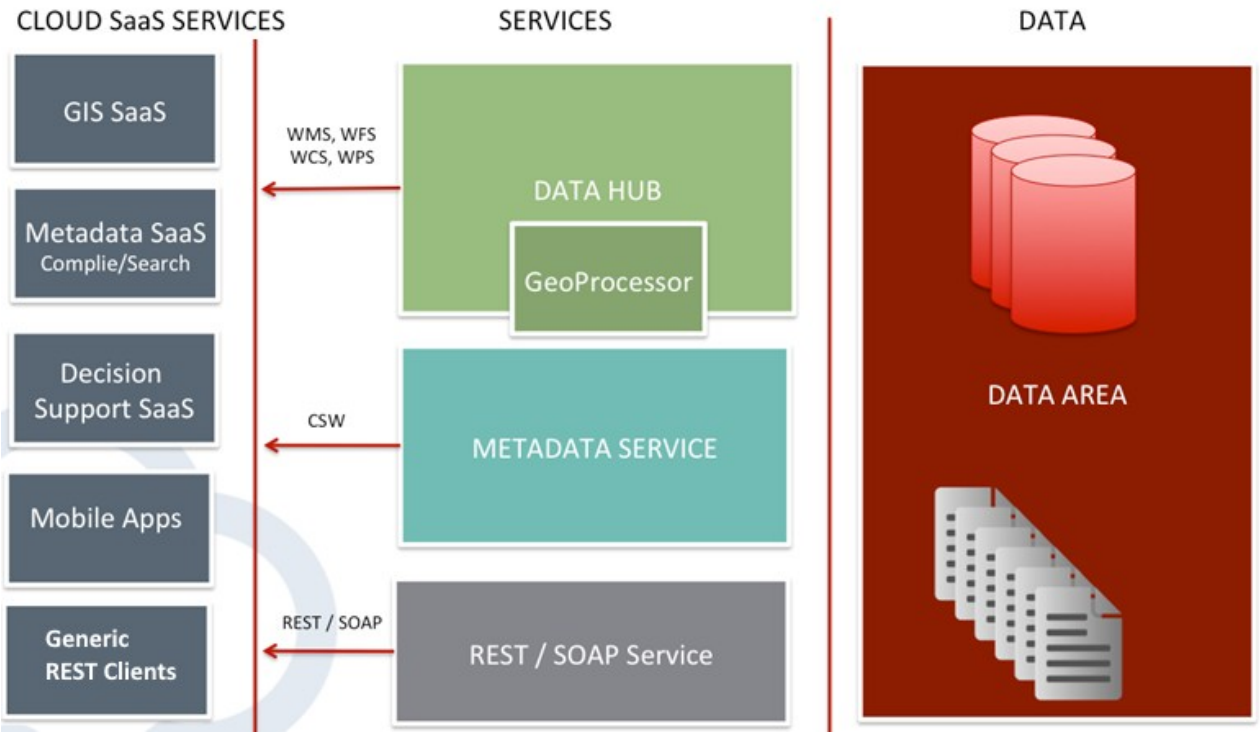


Esempio di gerarchie territoriali relazionate

La cartografia territoriale di base può essere in formato ESRI Shapefile o all'interno del geodatabase.

8.2.6 SPATIAL DATA INFRASTRUCTURE (GIS – SERVICE DATA HUB – DATA)

Una visione d'insieme di un sistema complesso che realizza la Spatial Data Infrastructure di i.TERPA è data dallo schema a blocchi logici presentato di seguito:



In questo schema di alto livello è mostrato come l'infrastruttura è visibile dall'esterno come un CLOUD SaaS Services (Software as a service).

La colonna a sinistra dell'immagine evidenzia l'insieme delle applicazioni fruibili dagli utenti o come servizi integrati in dei moduli applicativi, o come servizi consumabili da sistemi genera purpose di terze parti ad alta interoperabilità. Riepilogando:

- GIS SaaS: Piattaforma WebGIS per la creazione, consultazione, interrogazione e gestione di Progetti Mappa. Lavora mediante fruizione di servizi web OGC Compliant (WMS,WFS,WCS,CSW) e interagisce con l'area servizi.
- Metadata SaaS. applicazione per la compilazione dei metadati, presenta maschere di editing che consentono agli utenti autorizzati di alimentare l'insieme dei metadati. I metadati così costruiti saranno poi erogati come servizio OGC CSW (Catalog Service for Web) dall'Area Servizi.
- Decision Support System. Servizi che ritornano elaborazioni utili come supporto alle decisioni.
- Mobile App Center. Insieme dei software per la gestione partecipata di informazioni territoriali. Include le Apps per dispsitivi mobile le cloud interfaces che consentono il colloquio delle stesse Apps con il back-end Servizi/Dati.
- Generic REST / SOAP Client. I servizi seguono i protocolli e gli standard REST al fine di garantire la massima utilizzabilità anche da sistemi terzi, naturalmente compliant

La colonna al centro dell'immagine fa riferimento ai cosiddetti "SERVICES" composti da:

- **Data HUB**. Infrastruttura Scalabile per l'erogazione di servizi di mapping conformi agli standard dell' Open Geospatial Consortium (OGC®) WMS (Web Map Service), WFS (Web Feature Service), WCS (Web Coverage Service). Consente inoltre la Tematizzazione delle mappe geografiche erogate in WMS mediante l'applicazione dello standard SLD (Style Layer Description).
- **GeoProcessor**. Modulo che istanzia Web Process Services (WPS) ed implementa numerosi "Application Profile" (AP), profili di servizi di elaborazione di dati Informativi Territoriali. Il modulo GeoProcessor può essere integrato nel Data HUB per facilitare le interazioni con i dati da esso gestiti e permettendo di evitare di inviare l'intero dato da elaborare all'interno della richiesta di servizio.
- **Metadata Service**. Come già definito è il Componente Server responsabile della erogazione di servizi OGC® CSW con i Metodi GetCapabilities, DescribeRecord, GetRecordById, GetRecords e Transaction. Espone dunque servizio di discovery di dati attraverso consultazione di metadati, corrispondenti ai più comuni profili di metadati (FGDC - Federal Geographic Data Committee, Dublin Core, ISO - International Organization for Standardization, ISO 19115/19139 per dataset, ISO 19119/19139 per servizi, NAP - North American Profile, INSPIRE - INfrastructure for Spatial Information in Europe, dataset e servizi.)
- **REST / SOAP Interfaces**. Stack di servizi di supporto per le interrogazioni su banche dati geografiche. Implementa il protocollo SOAP (Simple Object Access Protocol) definendo una struttura dati per lo scambio di messaggi tra applicazioni. Utilizza HTTP come protocollo di trasporto, ma non è limitato nè vincolato ad esso, dal momento che può benissimo usare altri protocolli di trasporto. Implementa inoltre interfaccia RESTFull per mappare le tipiche operazioni CRUD(creazione, lettura, aggiornamento, eliminazione di una risorsa) e i metodi HTTP POST –> Create (Crea un nuova risorsa), GET –> Read (Ottiene una risorsa esistente), PUT –> Update (Aggiorna una risorsa o ne modifica lo stato), DELETE –> Delete (Elimina una risorsa)

Infine nella colonna destra, nell'immagine, si fa riferimento ad una infrastruttura unica per la memorizzazione di dati geografici (vettoriali e raster) e alfanumerici. Da qui sono aperte interfacce di connessione per lo strato SERVICE e si compone di due sub-infrastructures:

Geo-database. Implementato mediante DBMS PostgreSQL con estensione spaziale PostGIS consente la gestione di basi di dati territoriali.

File System Storage Area. Struttura File System dedicata alla memorizzazione di coperture raster. Implementabile come risorsa NFS (Network File System) accessibile dai server che implementano lo strato SERVICES, che necessitano di colloquiare con il dato RAW per la costruzione ed erogazione di servizi di mappa o di elaborazione.

8.3 PREPARAZIONE AMBIENTE INTER.PA

Al fine di una corretta implementazione dell'ambiente, di seguito sono descritte le modifiche da apportare per preparare le macchine alle installazioni.

8.3.1 Modifica file Hosts

Innanzitutto modificare il file host di tutte le macchine al seguente percorso:

```
vi /etc/hosts
```

Ed incollare all'interno del file il seguente contenuto:

Saranno così descritti tutti i puntamenti alle VM dell'ambiente

8.3.2 Java

In questo paragrafo sono descritti i passaggi per l'installazione e la configurazione di Java.

8.3.2.1 Versione

Si prevede l'installazione dei seguenti prodotti:

- JDK 1.8

8.3.2.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.2.3 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

Per il corretto funzionamento di Tomcat inizializziamo e definiamo i path necessari

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

8.3.2.4 Monitoraggio

- Per controllare che Java sia correttamente installato utilizzare i comandi:

```
java -version
```

- Output:

```
openjdk version "1.8.0_222"
OpenJDK Runtime Environment (build 1.8.0_222-b10)
OpenJDK 64-Bit Server VM (build 25.222-b10, mixed mode)
```

```
which java
```

- Output:

```
/bin/java
```

8.3.3 ActiveMQ

8.3.3.1 Versione

- ActiveMQ 5.14.5

8.3.3.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo Java*)

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.3.3 Installazione

Per l'implementazione di ActiveMQ è stato utilizzato il file apache-activemq-5.14.5.tar.gz

Spostare tramite protocollo SCP il suddetto file sulla macchina da configurare

- Spostare il file appena trasferito nella cartella /opt/:

```
mv apache-activemq-5.14.5.tar.gz /opt/
```

- Scompattare il file

```
tar xfvz apache-activemq-5.14.5.tar.gz
```

- Modificare l'ownership della cartella affidandola all'utente iterusr:

```
chown -R iterusr:iterusr apache-activemq-5.14.5/
```

- Creare il servizio per la gestione del software ActiveMQ

```
vi /etc/systemd/system/activemq.service
```

- Incollare quanto di seguito:

```
[Unit]
Description=HIP Message Broker
After=network.target

[Service]
Type=forking
WorkingDirectory=/opt/apache-activemq-5.14.5/bin
```

```
User=iterusr
Group=wheel
ExecStart=/opt/apache-activemq-5.14.5/bin/activemq start
ExecStop=/opt/apache-activemq-5.14.5/bin/activemq stop
Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64

[Install]
WantedBy=multi-user.target
```

- Salvare ed uscire con la sequenza:

```
:wq
```

- Avviare il servizio:

```
systemctl start activemq
```

8.3.3.4 Monitoraggio

Per controllare che il servizio sia attivo interrogare le porte che il software utilizza per le connessioni:

```
netstat -an | grep 1883
netstat -an | grep 8161
```

Se entrambe sono in stato "LISTEN" allora il servizio è correttamente avviato e attivo.

8.3.3.5 Comandi per la gestione del servizio

- Avvio del servio:

```
systemctl start activemq
```

- Stop del servizio:

```
systemctl stop activemq
```

- Check del servizio:

```
systemctl status activemq
```

- Restart del servizio

```
systemctl restart activemq
```

8.3.4 GFLEET

8.3.4.1 Versioni

L'installazione della gfleet-webapp prevede come requisiti i seguenti prodotti:

- JDK 1.8
- Tomcat 8.5.42
- Elastic Search 6.4.0

8.3.4.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java Installato (*vedi paragrafo java*)

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.4.3 Installazione Java

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel.x86_64
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

Per il corretto funzionamento di Tomcat inizializziamo e definiamo i path necessari.

- Modificare l'environment con il comando seguente:

```
vi /etc/environment
```

- Incollare all'interno quanto segue:

```
JAVA_HOME="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/"
```

- Salvare e uscire utilizzando la combinazione:

```
:wq!
```

- Assicurarsi di aver effettuato l'accesso come root e spostarsi nella cartella di root:

```
sudo su
cd
```

- Modificare il file di configurazione bash_profile:

```
vi .bash_profile
```

- Aggiungere alla fine di esso le seguenti righe:

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre/
export PATH=$JAVA_HOME/bin:$PATH
```

- Utilizzare di nuovo la combinazione per salvare ed uscire ed effettuare un refresh del file con il comando:

```
source .bash_profile
```

- Per verificare che il path sia giusto eseguire il comando seguente:

```
echo $JAVA_HOME
```

- Questo ci restituirà:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

8.3.4.4 Installazione Tomcat

Assicurarsi di aver effettuato l'accesso come utente root.

- Aggiungere gruppo e utente Tomcat per rendere più sicuri gli accessi:

```
groupadd tomcat
useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat
```

- Spostarsi nella directory che ospiterà Tomcat:

```
cd /opt
```

- Assicurarsi che il pacchetto wget sia installato:

```
yum install wget
```

- Effettuare il download di Tomcat dal repository ufficiale:

```
wget http://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.42/bin/apache-tomcat-8.5.42.tar.gz
```

- Scompattare il file:

```
tar -xzvf apache-tomcat-8.5.42.tar.gz
```

- Rinominare la cartella per facilitarne l'accesso:

```
mv apache-tomcat-8.5.42 tomcat/
```

- Spostarsi ora nella cartella bin di Tomcat per effettuare lo start del prodotto:

```
cd /home/centos/tomcat/bin
```

- Per avviare Tomcat utilizzare il comando seguente:

```
/home/centos/tomcat/bin
```

- I logs di avvio si possono reperire nel file di seguito:

```
cat /home/centos/tomcat/logs/catalina.out
```

- Se la procedura è andata a buon fine, collegandoci dal browser alle macchine ed utilizzando la porta di default di Tomcat, possiamo vedere la pagina iniziale di benvenuto:

```
http://10.240.10.37:8080
```

Copiare la cartella webapps del vecchio progetto nella nuova directory di Tomcat sostituendo quella di default.

- Modificare i permessi cambiando il proprietario della cartella nell'omonimo user:

```
chown -hR tomcat:tomcat tomcat
```

Creare il servizio di Tomcat in modo da rendere più veloce e semplici il monitoraggio.

- Creare il file:

```
vi /etc/systemd/system/tomcat.service
```

- Inserire all'interno di questi il seguente contenuto:

```
[Unit]
Description=Apache Tomcat Portal Server
After=network.target

[Service]
Type=forking
ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh
User=tomcat
Group=tomcat
LimitNOFILE=65536
```

```
[Install]
WantedBy=multi-user.target
```

- Cambiare owner ed assegnare permessi di execute al nuovo service:

```
chmod u+x /etc/systemd/system/tomcat.service
chown tomcat:tomcat /etc/systemd/system/tomcat.service
```

- Utilizzare ora il comando seguente per avviare tomcat:

```
systemctl start tomcat
```

8.3.4.5 Installazione ElasticSearch

Si proceda ora con l'installazione di ElasticSearch

- Perché funzioni tutto correttamente, creare un nuovo utente, che farà parte di un omonimo gruppo, che sarà specializzato nell'utilizzo di ElasticSearch:

```
groupadd elasticsearch
useradd elasticsearch -d /opt/elasticsearch-6.4.0/
```

- Copiare la cartella di ElasticSearch sotto il seguente path:

```
cp /home/centos/elasticsearch-6.4.0 /opt/ -rf
```

- Assegnare ownership e permessi alla directory appena copiata all'utente creato poco fa:

```
chown -hR elasticsearch:elasticsearch elasticsearch-6.4.0/
```

- Creare il file per il monitoraggio del servizio:

```
vi /etc/systemd/system/elasticsearch.service
```

- Incollare all'interno le seguenti righe:

```
[Unit]
Description=Elasticsearch
Documentation=http://www.elastic.co
Wants=network-online.target
After=network-online.target


[Service]
User=elastic
Group=elastic

ExecStart=/home/elastic/elasticsearch/bin/elasticsearch -p
/home/elastic/elasticsearch/run/elasticsearch.pid --quiet

# StandardOutput is configured to redirect to journalctl since
# some error messages may be logged in standard output before
# elasticsearch logging system is initialized. Elasticsearch
```

```
# stores its logs in /var/log/elasticsearch and does not use
# journalctl by default. If you also want to enable journalctl
# logging, you can simply remove the "quiet" option from ExecStart.
StandardOutput=journal
StandardError=inherit

# Specifies the maximum file descriptor number that can be opened by this
process
LimitNOFILE=65536

# Specifies the maximum number of bytes of memory that may be locked into RAM
# Set to "infinity" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml and 'MAX_LOCKED_MEMORY=unlimited' in ${path.env}
#LimitMEMLOCK=infinity

# Disable timeout logic and wait until process is stopped
TimeoutStopSec=0

# SIGTERM signal is used to stop the Java process
KillSignal=SIGTERM

# Java process is never killed
SendSIGKILL=no

# When a JVM receives a SIGTERM signal it exits with code 143
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

8.3.4.6 Monitoraggio

- Per controllare che la versione di Java sia quella giusta utilizzare il comando:

```
java -version
```

- Per controllare quale sia il path della JAVA_HOME utilizzare il comando:

```
echo $JAVA_HOME
```

- Verificare che sia uguale alla seguente:

```
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/jre/
```

- Per controllare che il servizio tomcat sia attivo utilizzare il comando di monitoraggio:

```
systemctl status tomcat
netstat -an | grep 8161
```

Se entrambe sono in stato “LISTEN” allora il servizio è correttamente avviato e attivo.

8.3.4.7 Comandi per la gestione del servizio

- Avvio del servio:

```
systemctl start activemq
```

- Stop del servizio:

```
systemctl stop activemq
```

- Check del servizio:

```
systemctl status activemq
```

- Restart del servizio

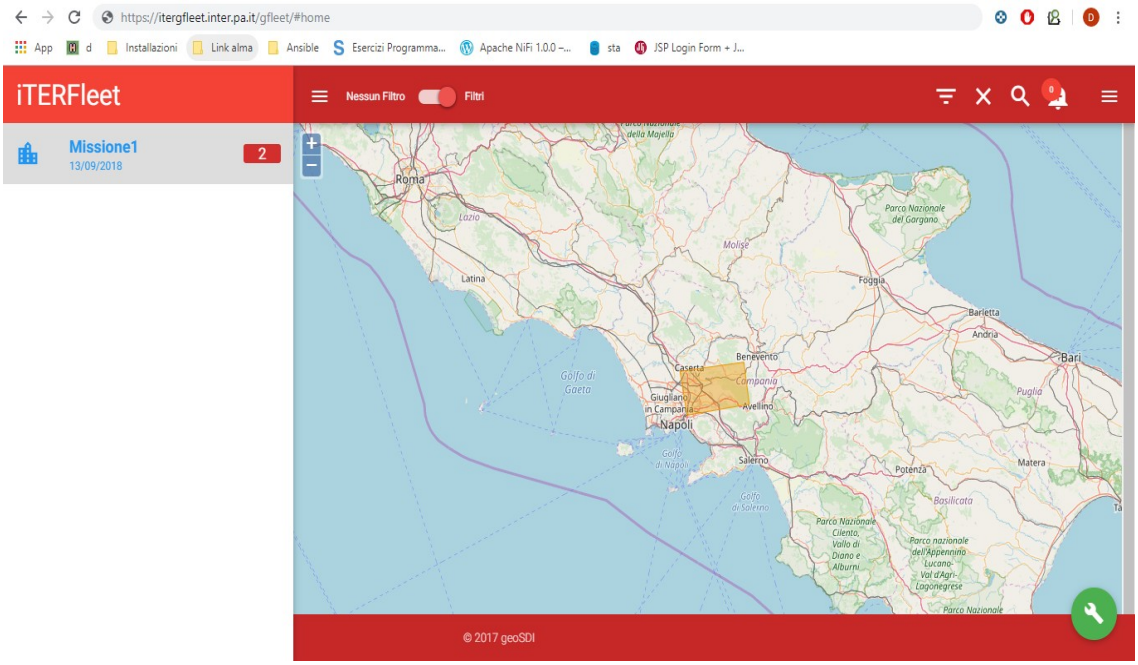
```
systemctl restart activemq
```

PROCEDURA PER RAGGIUNGERE LA PAGINA GFLEET

- Dal browser inserire l'URL dell'applicativo gfleet:

http://itergfleet.inter.pa.it/gfleet/

Pagina di Destinazione



8.3.4.8 Versioni

La macchina gfleet-service prevede l'installazione dei seguenti prodotti:

- JDK 1.8
- Tomcat 8.5.5

8.3.4.9 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati

Per la verifica del secondo punto eseguire il comando:

```
sudo yum update
```

8.3.4.10 Installazione

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
sudo yum install java-1.8.0-openjdk
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

8.3.4.11 Versioni

La macchina gfleet-websocket prevede l'installazione della versione 1.8 del prodotto JDK.

8.3.4.12 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati

Per la verifica del secondo punto eseguire il comando:

```
sudo yum update
```

8.3.4.13 Installazione

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
sudo yum install java-1.8.0-openjdk
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

8.3.4.14 Monitoraggio

- Per testare il corretto funzionamento della websocket, e dei suoi canali di comunicazione creare il seguente script:

```
vi test_ws.sh
```

- Incollare all'interno il seguente codice:

```
curl --include \
     --no-buffer \
     --header "Connection: Upgrade" \
     --header "Upgrade: websocket" \
     --header "Host: example.com:80" \
     --header "Origin: http://example.com:80" \
     --header "Sec-WebSocket-Key: SGVsbG8sIHdvcmxkIQ==" \
     --header "Sec-WebSocket-Version: 13" \
     https://10.244.10.39:9292/obu
```

- Fornire i permessi all'utente per l'esecuzione dello script:

```
chmod u+x test_ws.sh
```

- Avviare lo script con il comando:

```
./test_ws.sh
```

- In caso di errore apparirà un messaggio che ne indicherà il motivo. Es:

```
curl: (35) SSL received a record that exceeded the maximum permissible
length.
```

- In caso positivo apparirà un messaggio simile:

```
HTTP/1.1 101
Upgrade: websocket
Connection: upgrade
Sec-WebSocket-Accept: qGEgH3En71di5rrssAZTmtRTyFk=
Date: Wed, 11 Sep 2019 11:11:56 GMT
```

8.3.4.15 Versioni

La macchina gfleet-mqtt prevede l'installazione della versione 1.8 del prodotto JDK.

8.3.4.16 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati

Per la verifica del secondo punto eseguire il comando:

```
sudo yum update
```

8.3.4.17 Installazione

Eseguire i comandi elencati di seguito per effettuare le installazioni

- Eseguire il comando per verificare la presenza del pacchetto di cui abbiamo bisogno:

```
yum search java
```

- Procedere con l'installazione dopo aver trovato come risultato qualcosa di simile:

```
java-1.8.0-openjdk.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk.x86_64 : OpenJDK Runtime Environment 8
java-1.8.0-openjdk-accessibility.i686 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility.x86_64 : OpenJDK accessibility connector
java-1.8.0-openjdk-accessibility-debug.i686 : OpenJDK accessibility connector for packages with debug on
java-1.8.0-openjdk-accessibility-debug.x86_64 : OpenJDK 8 accessibility connector for packages with debug on
java-1.8.0-openjdk-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-debug.x86_64 : OpenJDK Runtime Environment 8 with full debug on
java-1.8.0-openjdk-demo.i686 : OpenJDK Demos
java-1.8.0-openjdk-demo.x86_64 : OpenJDK Demos 8
java-1.8.0-openjdk-demo-debug.i686 : OpenJDK Demos with full debug on
java-1.8.0-openjdk-demo-debug.x86_64 : OpenJDK Demos 8 with full debug on
java-1.8.0-openjdk-devel.i686 : OpenJDK Development Environment
java-1.8.0-openjdk-devel.x86_64 : OpenJDK Development Environment 8
java-1.8.0-openjdk-devel-debug.i686 : OpenJDK Development Environment with full debug on
java-1.8.0-openjdk-devel-debug.x86_64 : OpenJDK Development Environment 8 with full debug on
java-1.8.0-openjdk-headless.i686 : OpenJDK Runtime Environment
java-1.8.0-openjdk-headless.x86_64 : OpenJDK Headless Runtime Environment 8
java-1.8.0-openjdk-headless-debug.i686 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-headless-debug.x86_64 : OpenJDK Runtime Environment with full debug on
java-1.8.0-openjdk-javadoc.noarch : OpenJDK 8 API documentation
java-1.8.0-openjdk-javadoc-debug.noarch : OpenJDK 8 API documentation for packages with debug on
java-1.8.0-openjdk-javadoc-zip.noarch : OpenJDK 8 API documentation compressed in single archive
java-1.8.0-openjdk-javadoc-zip-debug.noarch : OpenJDK 8 API documentation compressed in single archive for packages with debug on
java-1.8.0-openjdk-src.i686 : OpenJDK Source Bundle
java-1.8.0-openjdk-src.x86_64 : OpenJDK Source Bundle 8
java-1.8.0-openjdk-src-debug.i686 : OpenJDK Source Bundle for packages with debug on
java-1.8.0-openjdk-src-debug.x86_64 : OpenJDK Source Bundle 8 for packages with debug on
```

- Installare, quindi, il pacchetto con il comando seguente:

```
sudo yum install java-1.8.0-openjdk
```

- Controllare che Java sia utilizzabile:

```
which java
```

- Quest'ultimo darà in output il path seguente:

```
/usr/bin/java
```

8.3.5 ElasticSearch

8.3.5.1 Versione

La versione prevista per l'installazione è:

- Elastic Search 5.4.1

8.3.5.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo java)*

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.5.3 Installazione Elastic Search

Per l'implementazione del software installare elasticsearch utilizzando il pacchetto della versione indicata in precedenza e modificando i file necessari.

Sono da modificare, nello specifico, la home directory e la directory in cui si trovano i file di configurazione di Elastic Search, rispettivamente (a sinistra il path in cui si possono consultare, a destra il nome del file compresso):

```
/usr/share/elasticsearch - elasticsearch-5.4.1.tar.gz
/etc/elasticsearch - etcelastic.tar.gz
```

- Dopo il trasferimento dei file -in formato .tar.gz- tramite protocollo SCP, scompattarli, a turno, e metterli nelle relative cartelle di destinazione:

```
tar xfvz elasticsearch-5.4.1.tar.gz
mv elasticsearch/ /usr/share/
```

- Ed ora l'altro file:

```
tar xfvz etcelastic.tar.gz
mv elasticsearc/ /etc/
```

- Per completare la configurazione, Elastic Search necessita di altre cartelle per il proprio funzionamento, utilizzare dunque i seguenti comandi per la creazione di esse:

```
mkdir /var/lib/elasticsearch
mkdir /var/run/elasticsearch
touch /etc/sysconfig/elasticsearch
```

- Elastic Search non può essere avviato da utente root per possibili problemi durante l'esecuzione. Creare dunque un utente addetto all'esecuzione del servizio garantendogli i permessi necessari:

```
groupadd elasticsearch
useradd elasticsearch -g wheel
usermod -aG wheel elasticsearch
```

- Cambiare ownership delle directory create prima:

```
chown root:elasticsearch /etc/elasticsearch/ -R
chown -R elasticsearch:elasticsearch /var/lib/elasticsearch/
chown -R elasticsearch:elasticsearch /var/run/elasticsearch/
chown -R elasticsearch:elasticsearch /etc/sysconfig/elasticsearch
```

- Creare ora il servizio attraverso il quale poter monitorare Elastic Search:

```
vi /etc/systemd/system/elasticsearch.service
```

- Incollare in esso il seguente contenuto:

```
[Unit]
Description=Elasticsearch
Documentation=http://www.elastic.co
Wants=network-online.target
After=network-online.target

[Service]
Environment=ES_HOME=/usr/share/elasticsearch
Environment=CONF_DIR=/etc/elasticsearch
Environment=DATA_DIR=/var/lib/elasticsearch
Environment=LOG_DIR=/var/log/elasticsearch
Environment=PID_DIR=/var/run/elasticsearch
EnvironmentFile=-/etc/sysconfig/elasticsearch

WorkingDirectory=/usr/share/elasticsearch

User=elasticsearch
Group=elasticsearch

ExecStartPre=/usr/share/elasticsearch/bin/elasticsearch-systemd-pre-exec

ExecStart=/usr/share/elasticsearch/bin/elasticsearch \
                                                    -p
${PID_DIR}/elasticsearch.pid \
                                                    --quiet \
                                                    -Edefault.path.logs=$
{LOG_DIR} \
                                                    -Edefault.path.data=$
{DATA_DIR} \
                                                    -Edefault.path.conf=$
```

```
{CONF_DIR}

# StandardOutput is configured to redirect to journalctl since
# some error messages may be logged in standard output before
# elasticsearch logging system is initialized. Elasticsearch
# stores its logs in /var/log/elasticsearch and does not use
# journalctl by default. If you also want to enable journalctl
# logging, you can simply remove the "quiet" option from ExecStart.
StandardOutput=journal
StandardError=inherit

# Specifies the maximum file descriptor number that can be opened by this
process
LimitNOFILE=65536

# Specifies the maximum number of bytes of memory that may be locked into RAM
# Set to "infinity" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml and 'MAX_LOCKED_MEMORY=unlimited' in
/etc/sysconfig/elasticsearch
LimitMEMLOCK=infinity

# Disable timeout logic and wait until process is stopped
TimeoutStopSec=0

# SIGTERM signal is used to stop the Java process
KillSignal=SIGTERM

# Java process is never killed
SendSIGKILL=no

# When a JVM receives a SIGTERM signal it exits with code 143
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target

# Built for distribution-5.4.1 (distribution)
```

- Dare maggiore disponibilità di memoria al servizio:

```
sysctl -w vm.max_map_count=262144
```

- Avviare ora il servizio:

```
systemctl start elasticsearch
```

8.3.6 WSO2ESB

8.3.6.1 Versione

La macchina prevede l'installazione della seguente versione del prodotto:

- Wso2esb 5.0.0

8.3.6.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo java)*

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.6.3 Installazione wso2esb

Per l'installazione del software wso2esb è stato utilizzato il pacchetto, fornito, di nome **wso2esb.tar.gz**.

Eseguire un trasferimento tramite protocollo SCP e una volta che il pacchetto è stato scaricato sulla macchina procedere come segue.

- Creare una cartella sotto il seguente path:

```
mkdir /opt/wso2esb-5.0.0/
```

- Muovere il file compresso nella cartella appena creata:

```
mv wso2esb.tar.gz /opt/wso2esb-5.0.0
```

- Scompattare il file:

```
tar xfvz wso2esb.tar.gz
```

- Creare il servizio per il software wso2esb al seguente path:

```
vi /etc/systemd/system/wso2.service
```

- Incollare in questo file quanto segue:

```
[Unit]
Description=WSO2 Enterprise Service Bus
After=syslog.target network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre
ExecStart=/opt/wso2esb-5.0.0/bin/wso2server.sh start
ExecStop=/opt/wso2esb-5.0.0/bin/wso2server.sh stop
#RemainAfterExit=yes
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target
Avviare il servizio:
systemctl start wso2esb
```

8.3.7 WSO2CEP

8.3.7.1 Versione

La macchina prevede l'installazione dei seguenti prodotti:

- Wso2cep 4.2.0

8.3.7.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo java)*

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.7.3 Installazione wso2cep

Per l'installazione del software wso2esb è stato utilizzato il pacchetto, fornito, di nome **wso2cep-4.2.0-bak.tar.gz**. Eseguire un trasferimento tramite protocollo SCP e una volta che il pacchetto è stato scaricato sulla macchina procedere come segue.

- Creare una cartella sotto il seguente path:

```
mkdir /opt/wso2cep-4.2.0/
```

- Muovere il file compresso nella cartella appena creata:

```
mv wso2cep-4.2.0-bak.tar.gz /opt/wso2cep-4.2.0
```

- Scompattare il file:

```
tar xfvz wso2cep-4.2.0-bak.tar.gz
```

- Creare il servizio per il software wso2cep al seguente path:

```
vi /etc/systemd/system/wso2cep.service
```

- Incollare in questo file quanto segue:

```
[Unit]
Description=WSO2 Complex Event Processor
After=syslog.target network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-
0.el7_6.x86_64/jre
ExecStart=/opt/wso2cep-4.2.0/bin/wso2server.sh start
ExecStop=/opt/wso2cep-4.2.0/bin/wso2server.sh stop
StandardOutput=syslog
StandardError=syslog

[Install]
WantedBy=multi-user.target
```

8.3.8 WSO2IS

8.3.8.1 Versioni

La macchina prevede l'installazione dei seguenti prodotti:

- MariaDB 5.5
- Wso2is 5.2.0

8.3.8.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo java)*

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.8.3 Installazione WSO2IS

- Creare una cartella sotto il seguente path:

```
mkdir /opt/wso2is/
```

- Muovere il file compresso nella cartella appena creata:

```
mv wso2is.tar.gz /opt/wso2is
```

- Scompattare il file:

```
tar xfvz wso2is.tar.gz
```

- Creare il servizio per il software wso2cep al seguente path:

```
vi /etc/systemd/system/wso2is.service
```

- Incollare in questo file quanto segue:

```
[Unit]
Description=WSO2 Enterprise Service Bus
After=syslog.target network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-
0.el7_6.x86_64/jre
ExecStart=/opt/wso2is/bin/wso2server.sh start
ExecStop=/opt/wso2is/bin/wso2server.sh stop
#RemainAfterExit=yes
StandardOutput=syslog
```

```
StandardError=syslog

[Install]
WantedBy=multi-user.target
```

8.3.8.4 Installazione MariaDB

Per l'installazione di MariaDB seguire i passi elencati.

- Scaricare il pacchetto:

```
yum install mariadb-server
```

- Avviare il servizio ed abilitarlo all'avvio del pc:

```
systemctl enable mariadb
systemctl start mariadb
```

- Effettuare la secure installation con il seguente comando:

```
mysql_secure_installation
```

Seguire i passi della procedura appena avviata e impostare la password per l'accesso come root e prenderne nota.

- Per effettuare l'accesso al db il comando sarà:

```
mysql -u root -p
```

- Inserire la password e premere invio

8.3.8.5 Import databases

Per l'import del database seguire pedissequamente i passi seguenti.

- Collegarsi alla macchina su cui è attivo il database al momento ed eseguire i seguenti comandi al fine di esportare tutti i databases:

```
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> CEP_GEO_DB.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> CEP_REG_LOCAL_1.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> CONFIG_DB_CEP.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> CONFIG_DB_ESB.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> CONFIG_DB_IS.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> ESB_REG_LOCAL_1.dmp
```

```
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> GOVERNANCE_DB.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> ITER_TEST.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> WSO2_USER_DB.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> hive_1.dmp
mysqldump -u root -p --routines --triggers --databases IDENTITY_REG_LOCAL_1
> IDENTITY_REG_LOCAL_1.dmp
```

- Creare sulla macchina in cui importare i db una cartella al percorso:

```
/home/centos/dump
```

- Trasferire tutti i dump appena creati nella cartella tramite protocollo SCP.
- Importare ora i database tramite i seguenti comandi dopo essersi spostati nella cartella dump:

```
cd /home/centos/dump
```

```
mysql -uroot -p CEP_GEO_DB < CEP_GEO_DB.dmp
mysql -uroot -p CEP_REG_LOCAL_1  < CEP_REG_LOCAL_1.dmp
mysql -uroot -p CONFIG_DB_CEP  < CONFIG_DB_CEP.dmp
mysql -uroot -p CONFIG_DB_ESB  < CONFIG_DB_ESB.dmp
mysql -uroot -p CONFIG_DB_IS  < CONFIG_DB_IS.dmp
mysql -uroot -p ESB_REG_LOCAL_1  < ESB_REG_LOCAL_1.dmp
mysql -uroot -p GOVERNANCE_DB  < GOVERNANCE_DB.dmp
mysql -uroot -p hive  < hive.dmp
mysql -uroot -p IDENTITY_REG_LOCAL_1  < IDENTITY_REG_LOCAL_1.dmp
mysql -uroot -p ITER_TEST  < ITER_TEST.dmp
mysql -uroot -p WSO2_USER_DB < WSO2_USER_DB.dmp
```

- Effettuare ora l'accesso al database ed eseguire i comandi successivi per creare nuovi utenti:

```
mysql -u root -p
```

```
CREATE USER 'hive'@'%' IDENTIFIED BY 'hive';
CREATE USER 'regadmin'@'%' IDENTIFIED BY 'regadmin';
```

- Eseguire i successivi comandi per garantire tutti i permessi del caso agli utenti ora creati:

```
GRANT ALL ON *.* TO regadmin@'%';
GRANT ALL ON *.* TO hive@'%';
```

8.3.9 Apache

8.3.9.1 Versione

La macchina prevede l'installazione del seguente prodotto:

8.3.9.2 Requisiti

Assicurarsi che sulla macchina ci siano i seguenti prerequisiti:

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Java installato (*vedi paragrafo java)*

Per la verifica del secondo punto eseguire il comando:

```
yum update
```

8.3.9.3 Installazione httpd

Per l'installazione del software procedere come descritto di seguito.

- Scaricare il pacchetto con il comando seguente:

```
yum install httpd
```

Per la configurazione di httpd sono stati usati gli stessi file utilizzati in CRED. Per cui seguire i passi elencati.

- Rimuovere ora le cartelle installate automaticamente dal gestore yum:

```
rm -rf /etc/httpd/conf.d/ /var/www/html
```

- Copiare tramite protocollo SCP le stesse cartelle da CRED a CLOUD le quali hanno già le configurazioni all'interno:

```
/etc/httpd/conf.d/
/var/www/html/
```

- Modificare tutti i file contenuti nella directory conf.d. Questi file contengono i puntamenti alle altre VM dell'ambiente. Sostituire perciò tutti gli indirizzi o i nomi host con i rispettivi che si trovano in CLOUD.
- Installare il pacchetto modulo ssl per il corretto funzionamento del software:

```
yum install mod_ssl
```

- Riavviare adesso il servizio rendendo effettive le modifiche:

```
systemctl restart httpd
```

8.3.10 PostgreSQL e PostGIS

8.3.10.1Versioni

- PostgreSQL 11
- PostGIS 2.5

8.3.10.2 Requisiti

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root

8.3.10.3 PostgreSQL

Per l'installazione di PostgreSQL procedere come di seguito.

Importare la chiave per l'accesso al repository:

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo
apt-key add -
```

Importare ora il repository:

```
RELEASE=$(lsb_release -cs)

echo "deb http://apt.postgresql.org/pub/repos/apt/ ${RELEASE}"-pgdg main |
sudo tee  /etc/apt/sources.list.d/pgdg.list
```

Aggiornare i repository:

```
sudo apt update
```

Procedere con l'installazione di PostgreSQL-11:

```
sudo apt -y install postgresql-11
```

Configurare il file postgresql.conf:

```
sudo vi /etc/postgresql/11/main/postgresql.conf
```

Cercare la voce listen_addressess e modificarla come illustrato:

```
listen_addresses = '*'
```

8.3.10.4 Login

Per effettuare il login eseguire i comandi in ordine:

```
sudo su
su - postgres
psql
```

8.3.10.5 Postgis

È un'estensione del database PostgreSQL, pertanto sarà integrato a quest'ultimo nel seguente modo.

Innanzitutto, installare il prodotto con il comando:

```
sudo apt install postgresql-11-postgis-2.5
```

Eseguire il login come utente postgres ed accedere al database:

```
sudo su
su - postgres
psql
```

Utilizzare ora i comandi elencati di seguito, uno alla volta, per aggiungere le estensioni:

- Enable PostGIS (includes raster)

```
CREATE EXTENSION postgis;
```

Per aggiungere nuove estensioni basterà utilizzare quindi il comando “CREATE EXTENSION” seguito dal nome dell’estensione desiderata.

8.3.11 Tomcat

8.3.11.1 Versioni

- PostgreSQL 11
- PostGIS 2.5

8.3.11.2 Requisiti

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l’accesso come root

8.3.11.3 Installazione Tomcat

Per l’installazione di Tomcat server eseguire i comandi elencati:

```
wget http://www-us.apache.org/dist/tomcat/tomcat-9/v9.0.20/bin/apache-tomcat-9.0.20.tar.gz
```

Scaricato il file estrarlo:

```
tar xzf apache-tomcat-9.0.20.tar.gz
```

Spostare poi la cartella estratta nel path preferito, nel caso seguente:

```
sudo mv apache-tomcat-9.0.20 /usr/local/apache-tomcat9
```

Creare l’utente ed il gruppo Tomcat per eseguire lo start e lo stop del servizio:

```
sudo groupadd tomcat

sudo useradd -s /bin/false -g tomcat -d /usr/local/apache-tomcat9 tomcat
```

Fornire i permessi utili al nuovo utente:

```
sudo chgrp -R tomcat /usr/local/apache-tomcat9/
```

Spostarsi nella home di Tomcat

```
cd /usr/local/apache-tomcat9/
```

Eseguire i seguenti cambi di permessi:

```
sudo chmod -R g+r conf
sudo chmod g+x conf
sudo chown -R tomcat webapps/ work/ temp/ logs/
```

Spostarsi nella cartella in cui creare il servizio di tomcat:

```
cd /etc/systemd/system/
```

Creazione del file del servizio di Tomcat:

```
sudo vi tomcat.service
```

Incollare all'interno del file i seguenti parametri:

```
[Unit]
Description=Apache Tomcat Portal Server
After=network.target
[Service]
Type=forking
ExecStart=/usr/local/apache-tomcat9/bin/startup.sh
ExecStop=/usr/local/apache-tomcat9/bin/shutdown.sh
User=tomcat
Group=tomcat
LimitNOFILE=65536
[Install]
WantedBy=multi-user.target
```

Salvare ed uscire e dare i permessi all'utente Tomcat per l'esecuzione del servizio:

```
sudo chmod u+x tomcat.service
```

Abilitare il servizio allo start della macchina ed avviarlo:

```
sudo systemctl enable tomcat
sudo systemctl start tomcat
```

8.3.12 Geoserver

8.3.12.1 Versioni

- Geoserver

8.3.12.2 Requisiti

- CentOS Linux release 7.4.1708
- Repository aggiornati
- Effettuare l'accesso come root
- Tomcat (*vedi paragrafo tomcat*)

8.3.12.3 Installazione Geoserver

Procedere come segue per l'implementazione del prodotto.

Creare una LVM proseguendo in questo modo:

```
sudo pvcreate /dev/xvdf
sudo vgcreate vg0 /dev/xvdf
```

Decidere ora la dimensione della memoria da allocare:

```
sudo lvcreate -l +100%FREE -n lv_portal  vg0
```

Nel caso precedente, allochiamo tutta la memoria (100%).

Per verificare la corretta implementazione utilizzare il comando:

```
sudo lvs
```

Procedere con la formattazione in XFS:

```
sudo mkfs.xfs /dev/mapper/vg0-lv_portal
```

Il prossimo comando serve ad estrapolare l'UUID:

```
sudo blkid /dev/mapper/vg0-lv_portal
```

Copiare l'UUID che sarà in questo formato:

```
UUID=417917c1-876b-4887-94fe-c3e799725463
```

Scrivere quest'ultimo nel file seguente in append in questo modo:

```
sudo vi /etc/fstab

UUID=417917c1-876b-4887-94fe-c3e799725463       /usr/local/apache-tomcat9
xfs     defaults        0       2
```

Eseguire il comando:

```
sudo mount -a
```

Verificare che il comando mount sia andato a buon fine:

```
sudo df | grep tomcat
```

A questo punto per l'installazione di Geoserver è necessario che sulla macchina sia installato, ed attivo, Tomcat. In caso Tomcat non sia installato seguire la procedura illustrata nei capitoli precedenti.

Modificare il limite dell'update di Tomcat modificando il file:

```
sudo vi /usr/local/apache-tomcat9/webapps/manager/WEB-INF/web.xml
```

Cercare la voce seguente e modificarla come segue:

```
<multipart-config>
      <!-- 50MB max -->
      <max-file-size>92428800</max-file-size>
```

```
      <max-request-size>92428800</max-request-size>
      <file-size-threshold>0</file-size-threshold>
    </multipart-config>
  </servlet>
  <servlet>
```

Garantire l'accesso all'admin modificando il file:

```
sudo vi /usr/local/apache-tomcat9/conf/tomcat-user.xml
```

Aggiungere tra i tag <tomcat-users … > e </tomcat-users> le seguenti righe:

```
<role rolename="manager-gui"/>
<user username="admin" password="S1nf1marzo2019" roles="manager-gui,admin-
gui,manager,admin,manager-script,admin-script"/>
```

Salvare ed uscire. Modificare ora il file context.xml per 'hardenizzare' tomcat:

```
sudo vi /usr/local/apache-tomcat9/conf/context.xml
```

Al momento non è stato fatto ma basterà modificare le righe seguenti:

```
  […] allow="127\.\d+\.\d+\.\d+|::1|0000:1|Indirizzo_Macchina" />
```

Scaricare il file geoserver.war dal link:

```
wget https://sourceforge.net/projects/geoserver/files/GeoServer/2.15.1/
geoserver-2.15.1-war.zip/download
```

Scaricare unzip:

```
apt install unzip
```

Utilizzare quest'ultimo per scompattare il file scaricato in precedenza di nome download:

```
unzip download
```

Copiare il file geoserver.war nel path seguente:

```
sudo cp geoserver.war /usr/local/apache-tomcat9/webapps/
```

Dare i permessi a Tomcat per questo file:

```
sudo chown tomcat:tomcat geoserver.war
```

9 GESTIONE DEGLI ASPETTI AMMINISTRATIVI DEL TRASFERIMENTO E GESTIONE DELLA BUONA PRATICA

Nell'ottica di poter mettere in pratica la Buona Pratica le amministrazioni su citate hanno stipulato una Convenzione che disciplina i rapporti tra le Parti per la realizzazione del Progetto "I.NTER.PA", nell'ambito della realizzazione degli obiettivi previsti dal PON Governance e capacità istituzionale 2014-2020.

La presente Convenzione definisce inoltre gli obblighi delle Parti, le procedure di rendicontazione e di pagamento.

Di seguito alleghiamo la suddetta Convenzione:

INTERPA_CONVENZI
ONE_REG_TOSCANA_E

Inoltre le parti hanno stipulato un Protocollo di Intesa che ha per oggetto la realizzazione in forma aggregata del progetto I.NTER.PA.

Il Protocollo regola quindi i rapporti tra gli Enti impegnati a realizzare il Progetto consentendo all'Ente capofila, che si conferma nella Regione Toscana, di assumere a propria volta tutti gli impegni derivanti dalla convenzione che esso stipulerà con l'Agenzia per la Coesione Territoriale.

Ai sensi dell'art. 2.3 dell'Avviso il progetto I.NTER.PA intende realizzare un intervento coerente con le seguenti linee operative:

- Miglioramento dell'efficacia e aumento dell'efficienza delle
- procedure a tutti i livelli dell'organizzazione amministrativa;
- Miglioramento della trasparenza, partecipazione e comunicazione a sostegno dell'azione amministrativa e Interoperabilità dei sistemi informativi con altre banche dati.

Gli Enti aderenti si impegnano a gestire in modo coordinato e sistemico la realizzazione del Progetto,

assicurando le migliori condizioni di efficienza e di economicità.

Il Protocollo di Intesa viene stipulato nell'ambito degli interessi istituzionali degli Enti partecipanti e ai fini dell'attuazione degli interventi e delle azioni necessarie all'attivazione dei prodotti e servizi.

Si allega di seguito il Protocollo di intesa stipulato con le parti:

Protocollo_intesa_tr
a_Partner PROG INTER