# MyStandard
# Specifiche di Architettura Tecnica

Versione 1.3

# SOMMARIO

## 2 RIFERIMENTI

| N. | Titolo | Autore | Versione | Data |
|---|---|---|---|---|
| | | | | |

## 3 GLOSSARIO

| Termine | Descrizione |
|---|---|
| | |

## 4 ACRONIMI

| Termine | Descrizione |
|---|---|
| | |

# 5 CONTESTO

Il Progetto MyStandard ha lo scopo di gestire Knowledge base semantiche secondo gli standard OWL / RDF.

L'obbiettivo del progetto è la realizzazione di un'applicazione che permetta:

- La definizione di un'ontologia di base ( catalogo di base ) con le entità di base per le ontologie di tutti domini di business
- La definizione di un'ontologia specifica sul dominio dei pagamenti.
- Di interrogare attraverso metadati, ricerche full text e ricerche semantiche delle interrogazioni sulle ontologie definite.
- La definizione di un processo di definizione di uno standard.
- L'integrazione ( opzionale ) con MyIntranet per permettere agli operatori degli enti di proporre / definire nuovi standard e con MyPortal per la pubblicazione dei cataloghi.

Lo scopo del presente documento è di illustrare la composizione architetturale della nuova applicazione MyStandard

# 6 ARCHITETTURA

Si descrive l'architettura della soluzione distinguendola sui tre livelli: funzionale, tecnologico e fisico.

## 6.1 FUNZIONALE

L'applicativo MyStandard nasce per la gestione di cataloghi e ontologie basati sugli standard OWL e RDF.

Il progetto prevede le seguenti macro funzionalità:

- Gestione CRUD dei Cataloghi di Base ( Ontologia di Base di MyStandard )
  - Entità Generica
  - Ente
  - Azienda ICT
  - Processi
  - API
- Gestione delle Entità specifiche del Mondo Pagamenti ( Ontologia Dominio Pagamenti )
- Implementazione delle funzionalità di Ricerca sul catalogo attraverso:
  - Ricerca per Metadati comuni a tutte le entità ( codice, versione, nome )
  - Ricerche Full Text
  - Ricerche Semantiche attraverso Query Semantiche ( SPARQL )
- Possibilità di definire e salvare per uso successivo un catalogo di Query Semantiche
- Integrazione (opzionale) di MyStandard con MyIntranet e MyPortal.
- Autenticazione e Autorizzazione tramite MyId e MyProfile. Implementazione delle funzionalità di definizione ed esecuzione di Query semantiche tramite SPARQL

Il progetto prevede l'organizzazione e la persistenza delle informazioni secondo gli standard semantici OWL e RDF.

### 6.1.1 Le Ontologie di MyStandard e OntoPIA

Uno degli obiettivi e dei requisiti del progetto MyStandard è la definizione di una o più ontologie con lo scopo di rappresentare al meglio i concetti semantici ( componente terminologica T- BOX ) e le entità ( componente asserzionale A-BOX) presenti nell'ecosistema della piattaforma MyPortal 3.

Nella modellazione concettuale e nello sviluppo delle Ontologie di MyStandard si farà riferimento all'iniziativa OntoPIA di AGiD il cui scopo è la definzione e la formalizzazione di un insieme di Ontologie a copertura dei concetti presenti nel mondo della PA.

In particolare:

- Le Ontologie di MyStandard utilizzano ed estendono alcuni concetti gia presenti nelle Ontologie di OntoPIA preservandone l'aderenza ai vocabolari controllati e agli standard DCAP-AT con profilo IT
  ( **https://ontopia-lode.agid.gov.it/lode/extract?url=https://w3id.org/italia/onto/DCAT#d4e1847**)
- Le Ontologie di MyStandard quando riutilizzano le Ontologie di OntoPIA sono sempre estensioni e non ridefiniscono la struttura sintattica e semantica dei concetti di OntoPIA. Non vi è quindi la necessità di applicare il validatore CPSV-AT sulle ontologie proprie definite da MyStandard.

## 6.2 TECNOLOGICO

È prevista la realizzazione di un nuovo applicativo denominato "MyStandard".

Il seguente schema architetturale evidenzia l'architettura della soluzione:


In particolare :

- La **nuova applicazione MyStandard** si compone di due moduli distinti:
    - **MyStandard Frontend** sviluppato con tecnologia Angular 10
    - **MyStandard Backend** sviluppato con tecnologia Java / SpringBoot.

      Per la gestione dei dati OWL e RDF il modulo di backend utilizzerà i framework open source:

        - Apache JENA per l'implementazione della logica di business e la manipolazione delle entità secondo lo standard RDF e per l'implementazione sei servizi di query con lo standard SparQL.

      La scelta di utilizzare il framework JENA deriva dal requisiti per cui la soluzione MyStandard deve gestire / interrogare / manipolare informazioni secondo gli standard propri nativi del web semantico ( RDF / OWL ) e le interrogazioni in formato SPARQL.

      Apache JENA è stato scelto in quanto
        - Il framework è disponibile con licenza open source
        - Dispone di API per la gestione semplificati degli standard RDF e OWL
        - E' potente e flessibile nell'implementazione di query semantiche con lo standard SPARQL.

    - I due moduli sono organizzati e deployati su un unico container Docker e il modulo di frontend è servito direttamente dall'applicazione SpringBoot

- L' Applicazione MyStandard **ha le seguenti dipendenze verso i seguenti servizi applicativi offerti da MyPlace**
    - **MyId:** Per l'autenticazione
    - **MyProfile :** Per la gestione dei Profili. L'istanza MyProfile utilizzata da MyStandard deve essere la medesima utilizzata dagli enti che utlizzano MyIntranet per la configurazione dei profili su MyProfile.
    - **Ceph (MyBox) :** Per la gestione degli allegati.
      MyStandard utilizza un bucket "dedicato" per la gestione degli allegati per cui può utilizzare anche un'istanza di MyBox separata dagli altri servizi MyPlace.
    - **Logstash ( MyLogs ) per la gestione dei log.**

      MyStandard utilizza SLF4J come libreria di Logging, configurata con un appender LogStash per raggiungere il servizio MyLogs.

      Può utilizzare il servizio MyLogs condiviso con MyPortal o un servizio MyLogs separato, purché sia disponibile la console di consultazione dei Log

- **A livello di servizi infrastrutturali** si evidenziano le seguenti dipendenze

- **Jena Fuseki2 TDB ( OWL / RDF Persistent Store ):** È la componente infrastrutturale core per i dati semantici e ontologici.

  Questa soluzione soddisfa tutti i requisiti richiesti:

  - E' una soluzione totalmente Free Open Source ( licenza Apache 2 )
  - La community è attiva e il progetto è gestito nella community Apache.
  - E' compatibile con gli standard RDF / OWL
  - Ha un motore SparQL 1.1
  - Supporta Jena e RDF4J ( attraverso Jena )
  - E' possibile installare il prodotto come:

    - Server Standalone
      https://jena.apache.org/documentation/fuseki2/fuseki-webapp.html#fuseki-standalone-server
    - Web Application
      https://jena.apache.org/documentation/fuseki2/fuseki-webapp.html#fuseki-web-application
    - Immagine Docker
      https://jena.apache.org/documentation/fuseki2/fuseki-main#fuseki-docker

- **ELK:** È una dipendenza diretta per le ricerche fulltext.
  MyStandard utilizza delle "collection" di ELK dedicate, per cui può utilizzare anche un'istanza di ELK separata dagli altri servizi MyPlace
- **Postgres :** È una dipendenza indiretta, che deriva dall'utilizzo di MyProfile
- **Mongo :** Per la gestione di alcuni metadati e la gestione di uno storico approvazioni legato alle "Entità" che non è persistito sullo store RDF

- Alcuni servizi REST devono essere esposti ed essere raggiungibili **dall'applicazione MyPortal.**

- **Apache Jena Fuseki2 con TDB2**

  Questa soluzione soddisfa tutti i requisiti richiesti:

  - E' una soluzione totalmente Free Open Source ( licenza Apache 2 )
  - La community è attiva e il progetto è gestito nella community Apache.
  - E' compatibile con gli standard RDF / OWL
  - Ha un motore SparQL 1.1

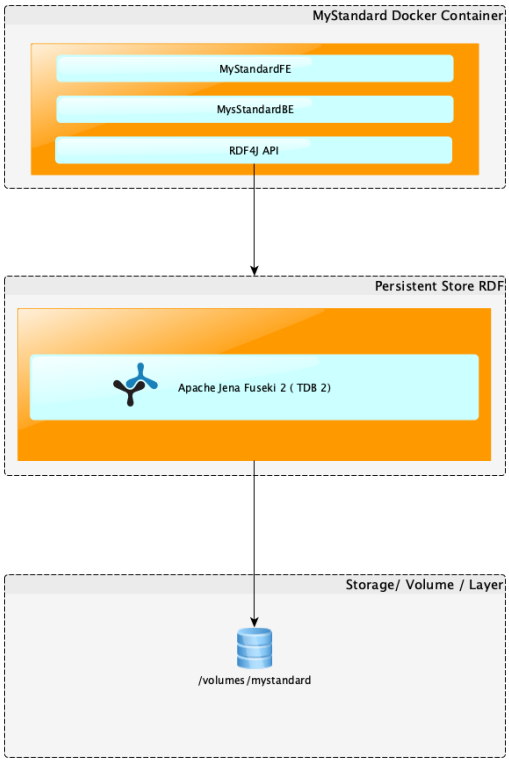
- Supporta Jena e RDF4J ( attraverso Jena )
- E' possibile installare il prodotto come:
    - Server Standalone
      https://jena.apache.org/documentation/fuseki2/fuseki-webapp.html#fuseki-standalone-server
    - Web Application
      https://jena.apache.org/documentation/fuseki2/fuseki-webapp.html#fuseki-web-application
    - Immagine Docker
      https://jena.apache.org/documentation/fuseki2/fuseki-main#fuseki-docker

Come sintesi finale dell'analisi comparativa  e considerando la matrice requisiti prodotti il prodotto Persistent Store migliore per MyStandard risulta essere:

### 6.3 FISICO

L'applicazione  MyStandard verrà fornito come un'unica immagine Docker.

L'applicazione MyStandard prevede l'utilizzo della soluzione Apache Jena Fuseki 2 con TDB2 ( https://jena.apache.org/documentation/fuseki2/ ) come Persistent Store RDF




con le seguenti note:

- L'installazione di Fuseki 2 può essere fatta in modalita: Standalone / Web / Docker.

- La componente Fuseki2 ed in particolare TDB2 **farà uso di un volume persistente in lettura e scrittura** ( alla stregua di altro prodotti DB ) per la quale è richiesto di applicare una politica di backup secondo le modalità preferite dalla struttura di Operations.

  Nel caso di installazione con Docker, il volume deve essere montato e reso disponibile in lettura e scrittura all'immagine Docker.

- Deve essere garantita la connettività di rete tra l'immagine di MyStandard e il server fuseki2.

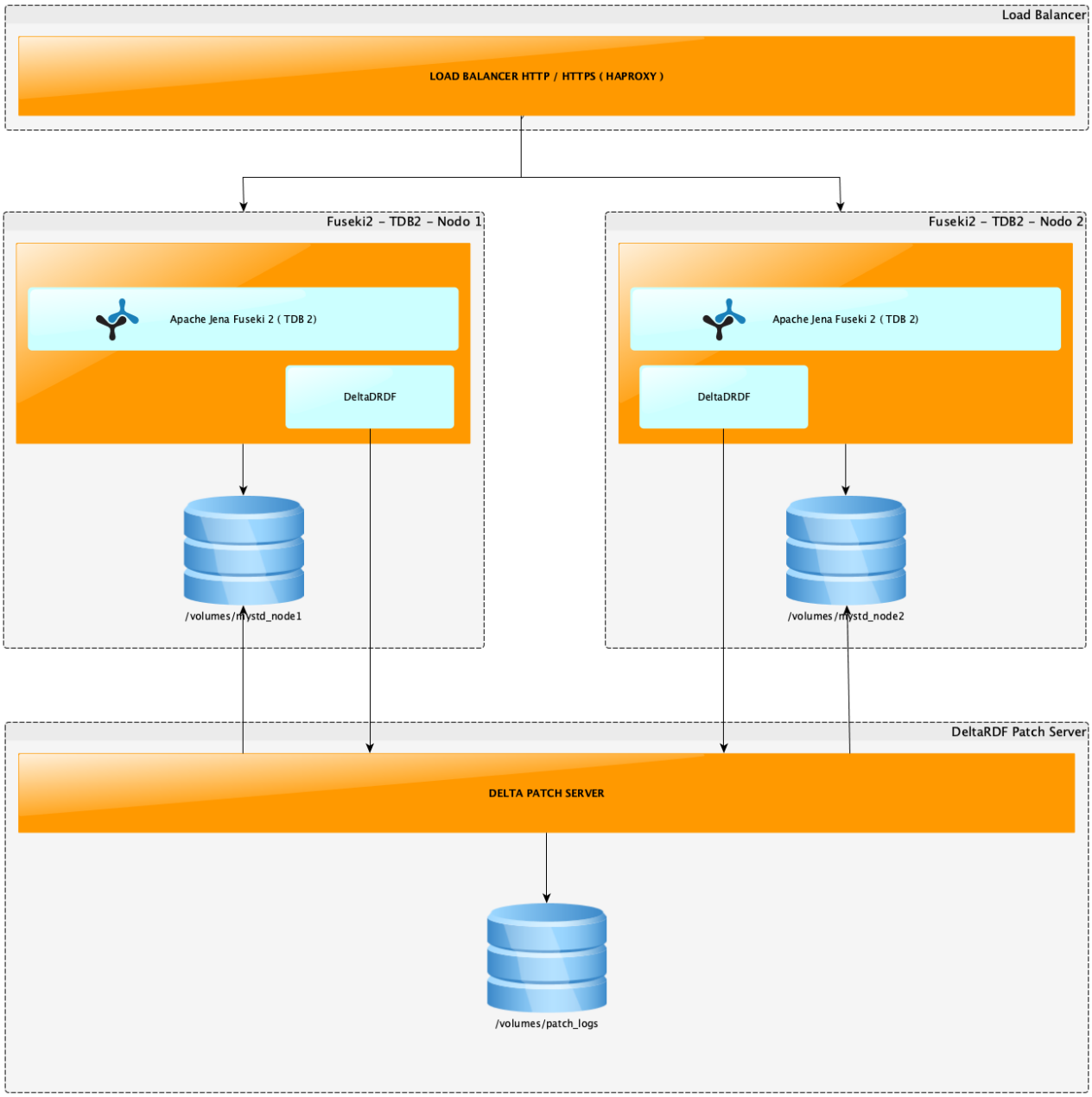
### 6.4 FUSEKI2 – TDB2 – HIGH AVAILABILITY

Come descritto nei paragrafi precedenti la soluzione Fuseki2 con TDB2 è stata scelta come Persistent Store per l'applicativo MyStandard in quanto soddisfa tutti i requisiti funzionali / tecnici e di dispiegamento richiesti.

Relativamente al dispiegamento installazione della componente Fuseki 2 sono disponbili tre diverse modalità:

1. Docker
2. Webapp
3. Standalone

Per l'installazione in produzione e garantire una soluzione in alta affidabilità è tuttavia consigliato utilizzare la modalità di dispiegamento di nodi multipli in modalità standalone di Fuseki 2 con il modulo Delta RDF ( https://afs.github.io/rdf-delta/ )




La soluzione per l'alta affidabilità prevede:

1. La presenza di n - nodi ( minimo 2 ) dove sia installato il prodotto Fuseki 2 TDB con le estensioni client del modulo Delta RDF ( https://repo1.maven.org/maven2/org/seaborne/rdf-delta/rdf-delta-dist/0.9.0/rdf-delta-dist-0.9.0.zip )

   Ognuno degli n nodi è attivo e ha un suo volume dedicato per i dati.
   Sugli n nodi viene installato il modulo RDF che è responsabile di propagare le notifiche al Delta RDF Server

2. La presenza di un bilanciatore di carico davanti agli n nodi Fuseki

3. La presenza di una componente Delta RDF Patch Server che riceve le notifiche di cambiamento dai nodi Fuseki2 e le propaga agli altri nodi. Nell'ottica dell'alta affidabilità va prevista anche l'installazione in alta affidabilità della componente Delta RDF Patch Server.

### 6.4.1 DELTA PATCH SERVER HIGH AVAILABILITY

Per la messa in affidabilità della componente Delta RDF Patch Server si ipotizza una soluzione basata su n nodi del patch server ( almeno 3 nodi ) coordinati da Apache Zookeper.

La soluzione è descritta qui:
https://afs.github.io/rdf-delta/ha-system.html#ha-patch-store

Sono previste le seguenti componenti per il Delta RDF Patch Server:

1. Almeno 3 nodi coordinanti da Apache Zookeper
2. L'uso di uno storage HA per il salvataggio del Patch Logs ( S3 Like, per esempio CEPH )