Allegato I. Architettura del Sistema Informativo Integrato FTWeb

CAPITOLATO TECNICO PRESTAZIONALE – Gestione, consolidamento ed evoluzione del Sistema Informativo Integrato (SII) di Forma.Temp

Sommario

1. Scopo e campo di applicazione del documento

1.1. Scopo del documento

Il documento ha come obiettivo la descrizione del sistema hardware e l'architettura di FTWEB.

Verranno illustrati in questo documento:

- Il disegno dell'architettura sistemistica della soluzione (HW);
- il dettaglio delle macchine utilizzate (HW).
- l'architettura logica della soluzione

1.2. Campo di applicazione

Le indicazioni contenute nel presente documento hanno validità per tutto il team di progettazione, sviluppo e test del corrente progetto

2. Documenti di riferimento e definizioni

2.1. Documenti di riferimento e definizioni

Codice / Titolo	Descrizione

2.2. Abbreviazioni

HW	Hardware
SW	Software
DB	Database
PG o Postgres	PostgreSQL
SSO	Single-Sign-On

3. Sistema hardware

3.1. Dettaglio delle risorse hardware

Il dettaglio di tutte le risorse hardware è descritto nel documento censimento Server (Allegato).

AssessmentCED.xlsx

.

Fare riferimento a quello relativamente a questo paragrafo. Tutte le macchine in ambito Forma.Temp sono macchine virtuali ad uso esclusivo. La gestione avviene tramite console VMware installato su cluster di 6 HP Server Proliant Gen8/9/10. Tutte le macchine, oltre all'utenza di root, hanno un'utenza di servizio (forma) con permessi di lettura soltanto in alcune cartelle specifiche (creata per permettere accesso in lettura al team di sviluppo dei files di logs nell'eventualità di velocizzare la risoluzione di anomalie gravi ed urgenti). La modalità di accesso avviene in ssh porta 22. Soltanto il personale SIA può anche accedere in console tramite client vmWare.

Sotto si riporta un dettaglio ulteriore sulle installazioni delle macchine.

macchina	IP interno	Note installazione
frm-cms-ese	192.168.110.56	• Installato dotCMS nella cartella /opt • E' raggiunto con mod_jk (ajp3) sulla porta 8009 (worker.properties) dall'entry point frm-httpd-ese (su quest'ultimo configurato solo sulla porta 80 in http)
frm-db-ese	192.168.110.57	• Installato PostGresql 9.2 • Il DB è in ascolto sulla porta 8000 in quanto la standard 5432 non era aperta tra la sede di Pomezia (produzione) e quella di Mirabello (gestione sistemi) • Gli schemi presenti sono frm_dotcms (portale istituzionale), frm_iam (sso) e frm_intranet (intranet)
frm-https-ese	192.168.110.58	• Apache configurato sia con mod_proxy e mod_jk • Configurato in ascolto sulla porta 80 e 443 (mod_ssl) • mod_jk à www.formatemp.it (porta 80) • mod_proxy à segnalazioni.formatemp.it, sso.formatemp.it e nuovaintranet.formatemp.it (porta 80 e 443); su questi ultimi è configurato il rewrite da 80 à 443
frm-redmine-ese	192.168.110.64	• Software installato Apache 2.4 (con Passenger), Redmine 4 e PostgreSql 12 • Apache in ascolto sulla porta 80 (DocumentRoot --> /var/www/redmine) • Postgresql 12 in ascolto sulla porta 5432 • Schema: redmine (ad uso di segnalazioni.formatemp.it)
frm-ftwebhttpd-ese	192.168.110.77	• Installato apache 2.4 con mod_proxy (in ascolto sulla porta 80 e 443) • Rewrite configurata: 80 --> 443 • Balancer con stickysession ON vs i 2 nodi di ftweb (frm-ftwebas1-ese ed frm-ftweb2-ese)
frm-ftwebas1-ese	192.168.110.78	• Installato WildFly 13 e Java JDk 1.8.0_172 • Su questa macchina accesso alla console di gestione cluster applicativo soltanto da rete interna e sulla porta 7580
frm-ftwebas2-ese	192.168.110.79	• Installato WildFly 13 e Java JDk 1.8.0_172
frmftwebdb1ese	192.168.110.91	• Installato PostgreSQL 12 e pgpool II 4 • DB: ftweb ; schema: formatemp_sii • Porta di ascolto 5433 • Installazione di pgpool con VIP 192.168.110.90 e porta 5432 • Streaming dei dati sul nodo secondario (frmftwebdb2ese) • Su questo nodo (primario) le connessioni stabilite sono in READ/WRITE

frmftwebdb2ese	192.168.110.92	• Installato PostgreSQL 12 e pgpool II 4 • DB: ftweb; schema: formatemp_sii • Porta di ascolto 5433 • Installazione di pgpool con VIP 192.168.110.90 e porta 5432 • Su questo nodo (secondario) le connessioni stabilite sono in READ
frm-intranet-ese	192.168.110.52	
frm-iam1-ese	192.168.110.53	

3.2.Configurazione dei sistemi

La mappatura dei servizi ospitati su ogni macchina è stata riportata nel documento Censimento Server. Sotto tabella con dettaglio di path files di configurazione e note sulla dimensione.

Macchina	path file di configurazione	files di configurazione	dimensione / crescita
frm-cms-ese			
frm-db-ese	/home/backups /root /var/lib/pgsql/data /lib/systemd/system	home/backups /pg_backup.config /home/backups /pg_backup_rotated.sh /home/backups /pg_backup_rotated.sh /root/.pgpass /var/lib/pgsql/data/pg_hba.conf /var/lib/pgsql/data/postgresql.conf /lib/systemd/system/postgresql.service	I DB attualmente hanno una dimensione di circa 8 MB compressi.
frm-httpd-ese	etc/httpd/conf /etc.httpd.conf.d /etc/httpd/certificati	etc/httpd/conf/httpd.conf etc/httpd/conf/worker.properties etc/httpd/conf.d/ssl.conf /etc/httpd/certificati/STAR_formatemp_it.ca-bundle /etc/httpd/certificati/STAR_formatemp_it.crt /etc/httpd/certificati/STAR_formatemp_it.crt	Occupazione disco con crescita di 2-3 GB anno (etc/httpd/log)
frm-redmine-ese	/home/backups /root	/home/backups /pg_backup.config	

	/etc/httpd /var/www/redmine/config var/lib/pgsql/12/data/	/home/backups /pg_backup_rotated.sh /home/backups /pg_backup_rotated.sh /root/.pgpass /etc/httpd/conf/httpd.conf /etc/httpd/conf.d/passenger.conf /var/www/redmine/config/routes.rb /var/www/redmine/config/configuration.yml /var/www/redmine/config/database.yml var/lib/pgsql/12/data/pg_hba.conf var/lib/pgsql/12/data/postgresql.conf	
frm-ftwebhttpd-ese	etc/httpd/conf /etc.httpd.conf.d /etc/httpd/certificati	httpd.conf ssl.conf rewriteHTTP.conf manutenzione.conf balancer.conf STAR_formatemp_it.ca-bundle STAR_formatemp_it.crt STAR_formatemp_it.crt	Occupazione disco con crescita di 2-3 GB anno (etc/httpd/log
frm-ftwebas1-ese	/opt/wildfly/bin /opt/wildfly/domain/configuration /opt/wildfly/domain/servers/ftweb1/configuration/FORMATEMP_SII	/opt/wildfly/bin/domain.conf /opt/wildfly/bin/add-user.sh /opt/wildfly/domain/configuration/mgmt-users.properties /opt/wildfly/domain/configuration/mgmt-groups.properties /opt/wildfly/domain/configuration/host-master.xml /opt/wildfly/domain/configuration/domain.xml /opt/wildfly/domain/servers/ftweb1/configuration/FORMATEMP_SII/applicationGlobal.properties /opt/wildfly/domain/servers/ftweb1/configuration/FORMATEMP_SII/carbon.properties	Attenzione ai files di log degli application server che sono di circa 600/700 MB giornalieri. Mensilmente sono spostati su storage di archiviazione.

frm-ftwebas2-ese	opt/wildfly/bin /opt/wildfly/domain/configuration /opt/wildfly/domain/servers/ftweb2/configuration/FORMATEMP_SII	opt/wildfly/bin/domain.conf opt/wildfly/bin/add-user.sh /opt/wildfly/domain/configuration/mgmt-users.properties /opt/wildfly/domain/configuration/mgmt-groups.properties /opt/wildfly/domain/configuration/host-master.xml /opt/wildfly/domain/configuration/domain.xml /opt/wildfly/domain/servers/ftweb1/configuration/FORMATEMP_SII/applicationGlobal.properties /opt/wildfly/domain/servers/ftweb1/configuration/FORMATEMP_SII/carbon.properties	Attenzione ai files di log degli application server che sono di circa 600/700 MB giornalieri. Mensilmente sono spostati su storage di archiviazione.
frmftwebdb1ese	/home/backups /root /var/lib/pgsql/.ssh /var/lib/pgsql/12/data/ /etc/pgpool-II/	/home/backups /pg_backup.config /home/backups /pg_backup_rotated.sh /home/backups /pg_backup_rotated.sh /root/.pgpass /var/lib/pgsql/.ssh/authorized_keys /var/lib/pgsql/.ssh/id_rsa /var/lib/pgsql/.ssh/id_rsa.pub /var/lib/pgsql/.ssh/known_hosts /var/lib/pgsql/12/data/ph_hba.conf /var/lib/pgsql/12/data/postgresql.conf /var/lib/pgsql/12/data/ph_hba.conf/recovery_1st_stage /etc/pgpool-II/pcp.conf /etc/pgpool-II/pgpool.conf /etc/pgpool-II/pool_hba.conf /etc/pgpool-II/pool_passwd /etc/pgpool-II/failover.sh	Il dump del DB compresso allo stato attuale è di circa 6GB, senza compressione circa 31 GB. Non è possibile fare una stima della crescita. Si riporta che il DB compresso a settembre 2019 era di circa 2 GB (10 GB non compressi).
frmftwebdb2ese	home/backups /root	/home/backups /pg_backup.config	dump del DB compresso

	/var/lib/pgsql/.ssh /var/lib/pgsql/12/data/ /etc/pgpool-II/	/home/backups /pg_backup_rotated.sh /home/backups /pg_backup_rotated.sh /root/.pgpass /var/lib/pgsql/.ssh/authorized_keys /var/lib/pgsql/.ssh/id_rsa /var/lib/pgsql/.ssh/id_rsa.pub /var/lib/pgsql/.ssh/known_hosts /var/lib/pgsql/12/data/ph_hba.conf /var/lib/pgsql/12/data/postgresql.conf /var/lib/pgsql/12/data/ph_hba.conf/recovery_1st_stage /etc/pgpool-II/pcp.conf /etc/pgpool-II/pgpool.conf /etc/pgpool-II/pool_hba.conf /etc/pgpool-II/pool_passwd /etc/pgpool-II/failover.sh	allo stato attuale è di circa 6GB, senza compressione circa 31 GB. Non è possibile fare una stima della crescita. Si riporta che il DB compresso a settembre 2019 era di circa 2 GB (10 GB non compressi).
frm-intranet-ese			
frm-iam1-ese			

Tabella procedure servizi (N.B. Tutti i servizi indicati sono abilitati allo start in automatico al riavvio della macchina)

Macchina	Path installazione servizio	Procedura/comando start/stop	NOTE
frm-cms-ese	/opt/dotcms	service dotcms start/stop	
frm-db-ese	/var/lib/pgsql	service postgresql start/stop	Il DB è in ascolto sulla porta 8000 .Il change port è stato fatto qui : /lib/systemd/system/postgresql.service
frm-httpd-ese	/etc/httpd	service httpd start/stop	
frm-redmine-ese	Apache: /etc/httpd Redmine: /var/www/redmine	Avviare prima DB e poi apache: service postgresql-12 start/stop service httpd start/stop	

	Postgresql 12: /var/lib/pgsql/12		
frm-ftwebhttpd-ese	/etc/httpd	service httpd start/stop	
frm-ftwebas1-ese	/opt/widlfly/ /opt/jdk1.8.0_172	service wildfly start/stop	Avviare wildfly prima su nodo 1 (ftwebas1) e poi 2 (ftwebas2)
frm-ftwebas2-ese	/opt/widlfly/ /opt/jdk1.8.0_172	service wildfly start/stop	
frmftwebdb1ese (master)	/var/lib/pgsql/12 /etc/pgpool-II	Service postgresql-12 start/stop Service pgpool start/stop	Master e slave in ascolto singolarmente sulla porta 5433. Nodo VIP in ascolto sulla 5432. start postgresql-12 start/stop start pgpool start/stop Corretta procedura di avvio: 1) start postgres slave 2) start postgres master 3) start pgpool slave 4) start pgpool master
frmftwebdb2ese (slave)	/var/lib/pgsql/12 /etc/pgpool-II	Service postgresql-12 start/stop Service pgpool start/stop	
frm-intranet-ese	/opt/dotcms	service dotcms start/stop	
frm-iam1-ese	/opt/wso2	service wso2 start/stop	

3.3.Caratteristiche infrastruttura di rete

Si riassumono le caratteristiche delle reti coinvolte nell'esercizio dei sistemi raggruppandole secondo tre tipologie: SAN – LAN – WAN.

SAN: ciascun host fisico è connesso tramite due link in fibra 8 Gbps attestati a due fabric separati, ognuno dei quali acceduto da uno Storage Processor del dispositivo di memorizzazione principale.

Il multipath I/O garantisce la piena continuità del servizio a fronte del guasto di una qualunque delle componenti sopracitate.

LAN: Ciascun host fisico è connesso allo switch modulare (Centro Stella) tramite link Etherchannel che sfrutta almeno due cavi in rame ad 1 Gbps.

Il throughput garantito dal Centro Stella è superiore a 48 milioni di pacchetti per secondo con velocità del fabric switch fino a 76,8 Gbps.

WAN: La connettività a Internet è garantita da una coppia di firewall attestati tramite link ad 1 Gbps alla dorsale TIM a 7,5 Gbps.

In nessuna delle tre tipologie di collegamenti sono presenti meccanismi di limitazione di banda o di reservation della stessa, ma tutti gli accessi vengono sempre effettuati in best effort

3.4.Politiche di sicurezza

Le politiche di sicurezza Eustema, conformemente alla norma internazionale ISO27001 sui Sistemi di Gestione della Sicurezza delle Informazioni, su cui l'azienda è certificata dal 2014, indirizzano tutti gli aspetti che possono

avere un impatto sulla sicurezza informatica. In estrema sintesi, vengono contemplate tutte le adeguate misure di sicurezza atte a garantire un utilizzo corretto e sicuro delle risorse informatiche e dei relativi dati: le risorse informatiche devono essere protette da controlli di accesso e da profilazione in base all'utente, al ruolo e alle attività nelle quali è impegnato; le utenze devono essere abilitate all'inizio del rapporto lavorativo e disabilitate al cessare dello stesso; le procedure di assunzione del personale devono prevedere accordi di riservatezza la cui validità deve estendersi oltre la cessazione del rapporto di lavoro; le risorse più sensibili devono essere protette da meccanismi di Strong Authentication e da cifratura; i portatili aziendali devono essere dotati di software di protezione antivirus e di cifratura; i server in ambiente di esercizio devono essere sottoposti a monitoraggio sia prestazionale che di sicurezza, il perimetro aziendale deve essere protetto con misure di sicurezza atte all'identificazione e blocco delle minacce; il personale deve essere adeguatamente formato, in base al ruolo e alle mansioni specifiche e devono essere periodicamente effettuate comunicazioni di security awareness rivolte a tutto il personale. I dati in transito e gli accessi remoti devono essere protetti da cifratura.

Sono inoltre attive:

- Piattaforma di security event management
  Piattaforma di centralizzazione dei log, atta al monitoraggio delle attività e degli accessi ai server. La piattaforma acquisisce e mantiene in modo inalterabile tutti i log dei server, consentendo di effettuare tutte le analisi e le correlazioni del caso.

- Piattaforma di breach protection
  Piattaforma di sicurezza allo stato dell'arte, atta all'identificazione e blocco dei breach, tramite un monitoraggio continuo di tutte le entità e le attività, indirizzando i punti deboli e le superfici di attacco esposte nell'intero ambiente.

3.5.Policy di sicurezza server

Le politiche di backup prevedono il backup quotidiano dell'intera macchina virtuale e la copia su sito remoto (distanza > 30 Km) per finalità di data recovery / disaster recovery.

Per il sito di esercizio è prevista una retention di almeno 90 giorni; nel sito remoto vengono mantenute le copie degli ultimi due backup effettuati con successo presso il sito di esercizio.

Come supporti di memorizzazione vengono utilizzate appliance di storage con HHDD di tipo SAS per il sito di esercizio (RAID 5) e di tipo SATA (RAID 5) per il sito di disaster recovery.

Per le sole macchine aventi un DB (frm-redmine-ese, frm-db-ese e nodo master frmftwebdb1ese) sono previsti in locale i backup di tutti gli schemi presenti. Tali backup sono schedulati in crontab e vengono completati prima che inizia il backup dell'intera macchina. I backup sono eseguiti tramite esecuzione dello script /home/backup/pg_backup_rotated.sh e depositati nella cartella /home/backups/database/postgresql/aaaa-mm-dd-daily

3.6.Policy di sicurezza dell'infrastruttura

Il sito di esercizio è allocato presso la struttura IDC di TIM, certificata ISO27001 e fornisce tutti le misure di sicurezza fisica necessarie (accessi monitorati, sorveglianza armata, continuità elettrica, condizionamento, antincendio ecc. ecc).

La ridondanza dei sistemi in tutte le diverse componenti (alimentatori, dischi, switch di rete, server…) elimina la presenza di single point of failure che possano rendere indisponibili i servizi.

La sicurezza logica è garantita da apparati "Unified Threat Management" che forniscono protezione a livello di rete (firewalling), applicativo (IPS) e di malware.

Da un punto di vista operativo le procedure di backup implementate permettono di ripristinare i sistemi nell'arco di minuti, e/o di renderli disponibili in modalità offline per eventuali verifiche di coerenza dei dati.

I dati sottoposti a backup sono cifrati e senza la pasword di decifratura ne è possibile il ripristino solo su sistemi riconosciuti dal sistema che ha effettuato il salvataggio del dato.

Il sito remoto di conservazione dei backup è inoltre predisposto perché possa divenire nell'arco qualche ora il sito di erogazione del servizio in caso di eventi disastrosi che abbiano reso completamente inagibile il sito primario

4. Architettura Logica della soluzione

Si riporta in questo capitolo lo schema raffigurante l'architettura logica della soluzione.

Per quanto riguarda invece l'utilizzo di librerie e/o framework si fornisce di seguito la lista completa

JAVA 8

JAVA-EE 7

eaeam-abstract-framework

jackson-jaxrs-json-provider v.2.7

JPA -- hibernate-core v 5.1

eaeam-client-auriga v 0.0.7

jasperreports v 6.8

itext v2.1.7

JAX-RS -- jboss resteasy v 3.0.14

apache.poi v 3.17

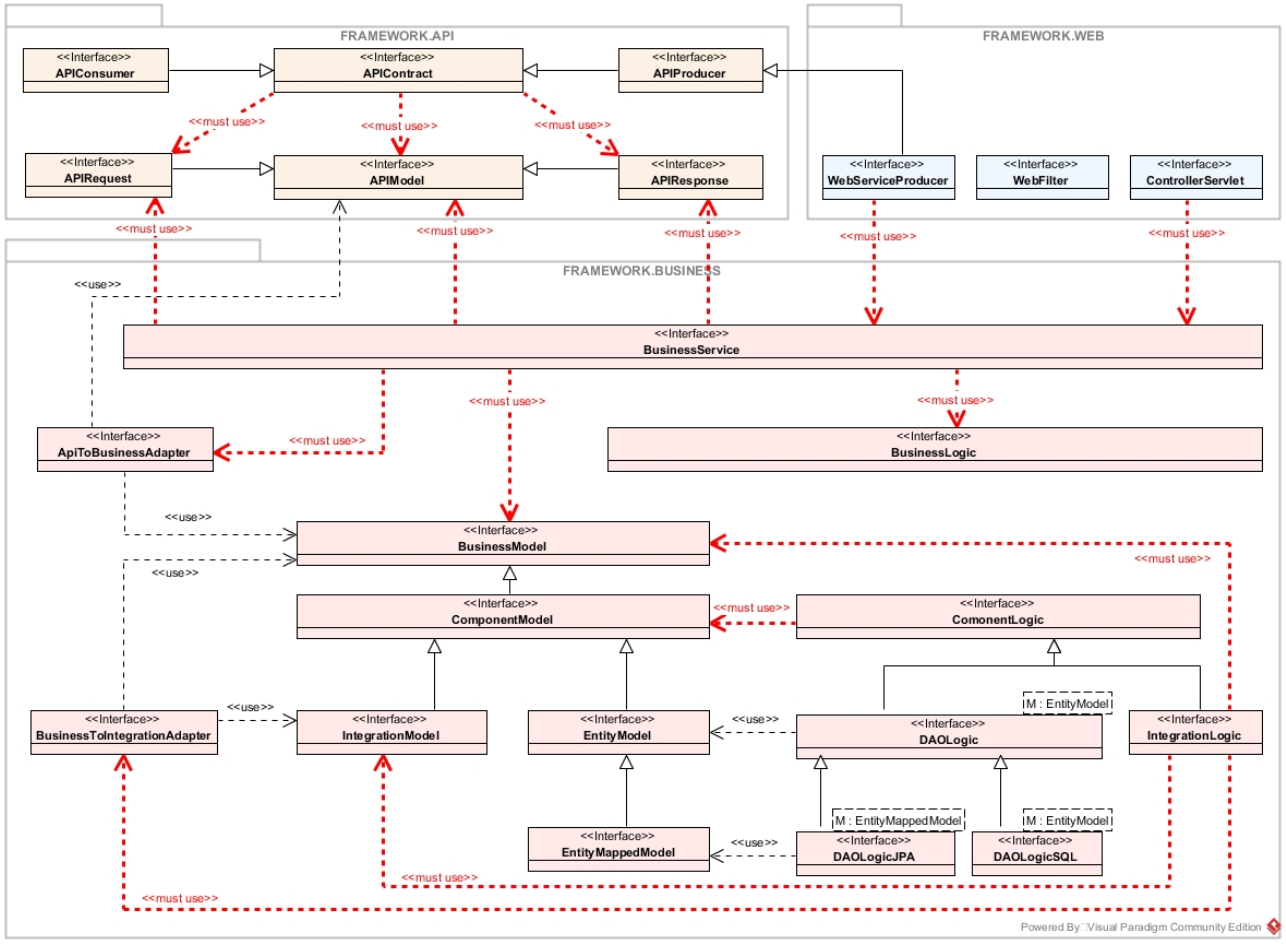
eaeam-client-wso2is v 0.0.3

I component che iniziano con **eaeam** fanno parte di un framework sviluppato direttamente da Eustema.

4.1.Disegno dell'architettura logica della soluzione

L'architettura di ftweb fa parte della categoria delle architetture dette a "servizi", ogni servizio al suo interno è composto da una serie di "strati" tecnici che permettono di avere un alto disaccoppiamento tra i modelli dati di business e i modelli di presentazione.

Di seguito è presente uno schema a blocchi dove vengono messi in evidenza i vari strati con il modello dati associato ad ogni strato.



Esistono quattro categorie diverse per il modello dati.

- **API MODEL REQUEST/RESPONSE**
- **API MODEL**
- **BUSINESS MODEL**
- **INTEGRATION MODEL**

Le prime due sono preposte alla presentazione del dato, ovvero contengono solo le informazioni necessarie alla presentazione del dato.

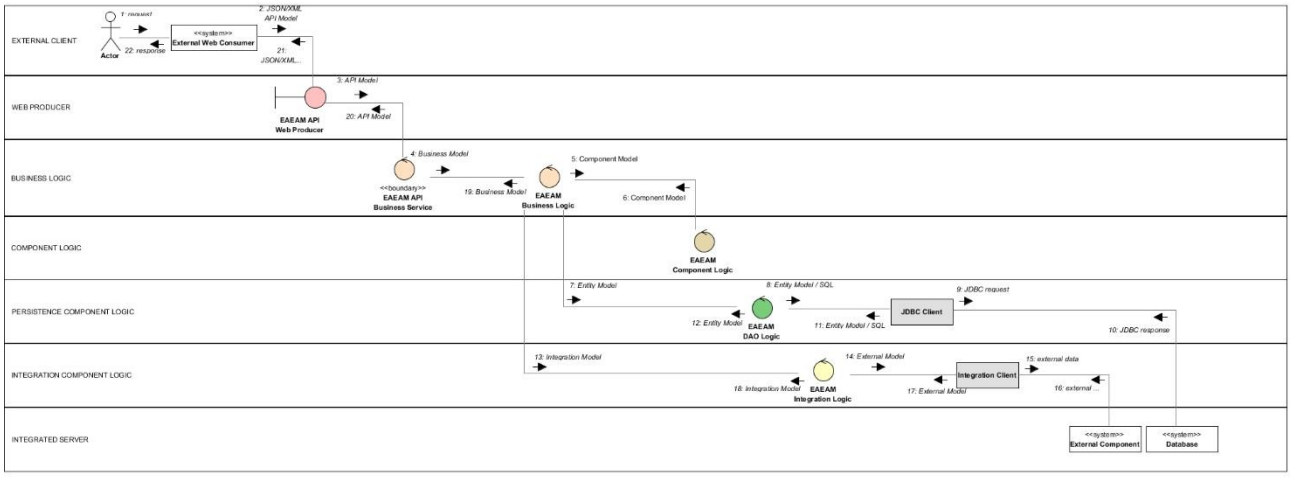
Il business model rappresenta il dato completo nella sua struttura e viene manipolato e/o costruito dagli strati tecnici di business logic per eseguire appunto le logiche previste dall'analisi funzionale.

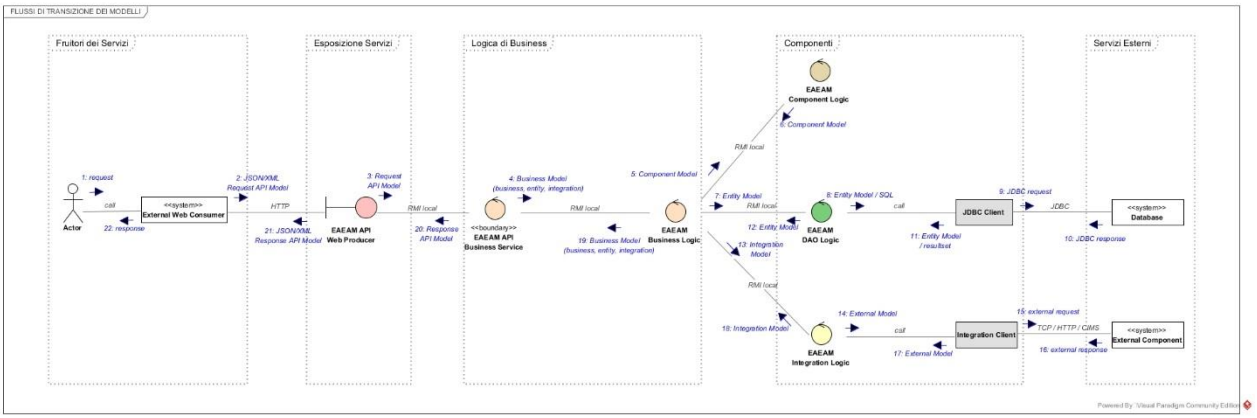
L'integration Model è il tipo di dato manipolato e/costruito negli strati di integration logic, tale strato tecnico ha il compito di preparare il modello in modo che sia fruibili dai servizi esterni ad ftweb.

In pratica l'integration logic è strato con cui ftweb comunica con i servizi e/o componenti esterni, come ad esempio WSO2 e Auriga.

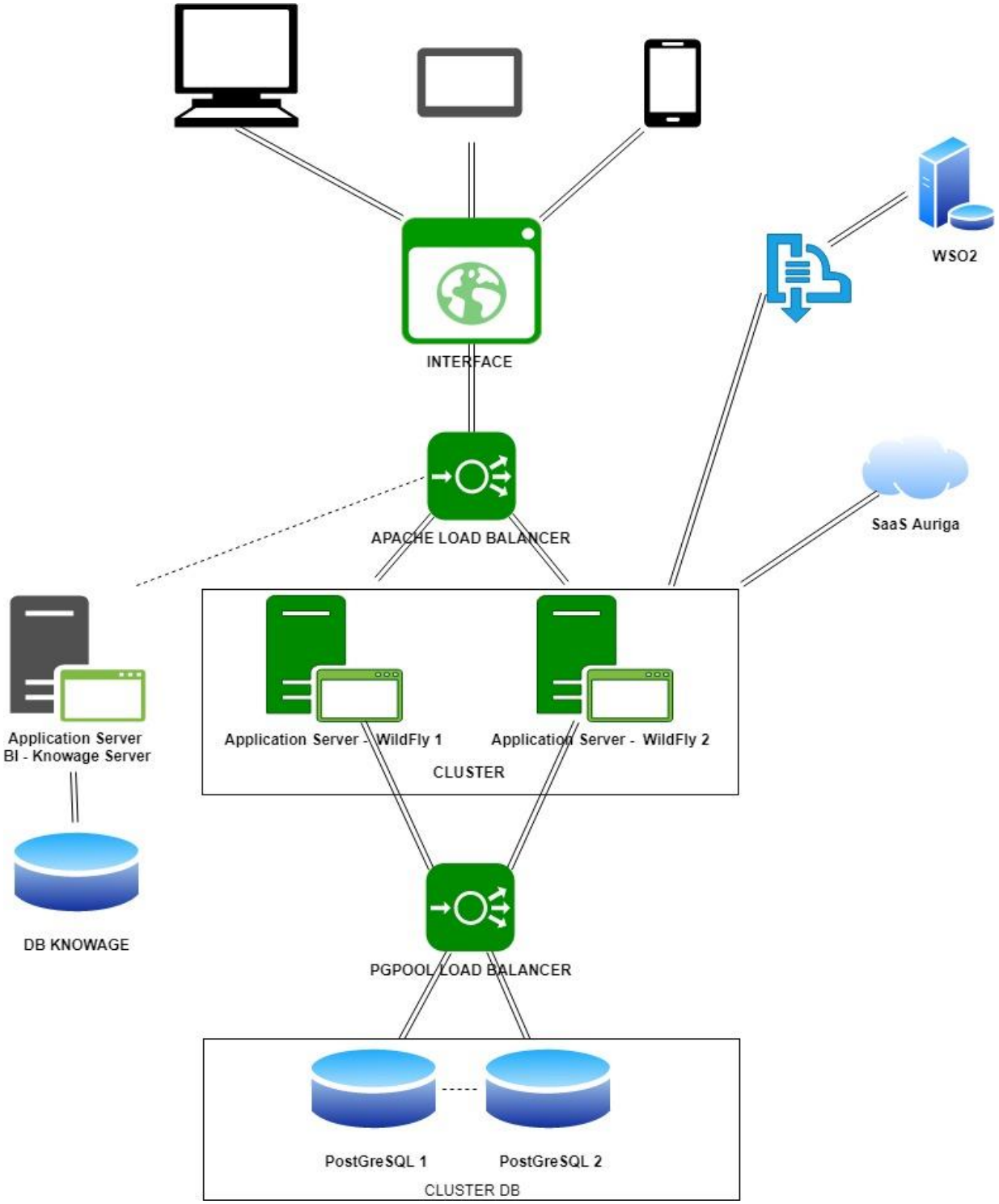
Questo strato consente dunque di mantenere separate le logiche di business dalle logiche necessarie all'interazione con i sistemi esterni.

Di seguito si mostra uno schema astratto riguardante la modalità con cui i dati viaggiano all'interno degli strati software.


EXTERNAL CLIENT
WEB PRODUCER
BUSINESS LOGIC
COMPONENT LOGIC
PERSISTENCE COMPONENT LOGIC
INTEGRATION COMPONENT LOGIC
INTEGRATED SERVER
Actor
<<system>>
External Web Consumer
EAEAM API
Web Producer
<<boundary>>
EAEAM API
Business Service
EAEAM
Business Logic
EAEAM
Component Logic
EAEAM
DAO Logic
JDBC Client
EAEAM
Integration Logic
Integration Client
<<system>>
External Component
<<system>>
Database


FLUSSI DI TRANSIZIONE DEI MODELLI
Fruitori dei Servizi
Esposizione Servizi
Logica di Business
Componenti
Servizi Esterni
Actor
<<system>>
External Web Consumer
EAEAM API
Web Producer
<<boundary>>
EAEAM API
Business Service
EAEAM
Business Logic
EAEAM
Component Logic
EAEAM
DAO Logic
JDBC Client
EAEAM
Integration Logic
Integration Client
<<system>>
Database
<<system>>
External Component

Di seguito abbiamo invece uno schema più ad alto livello della soluzione e che comprende anche i componenti esterni.



Il nodo PostGreSQL1 del Cluster DB consente sia la lettura che la scrittura dei dati su DB, mentre il nodo 2 consente la sola lettura del dato.

4.1.1. Presentazione

Le interfacce che costituiscono il front-end di ftweb sono state sviluppate utilizzando i seguenti framework e linguaggi.
HTML 5
CSS 3
Angular 6 – typescript

4.1.1.1. Servizi REST

Il sistema espone dei servizi REST che utilizzano strutture dati in formato JSON (stringhe di informazioni) per l'interscambio dei dati nelle sessioni di browsing verso gli utenti e per l'interlocuzione con i servizi offerti in modo diretto alle ApL.

Il JSON in input viene prodotto

- da client REST, per quanto riguarda i Web services delle ApL
- dal framework Angular 6, per le sessioni di navigazione ed utilizzo della piattaforma via browser sui client degli utenti
    - L'utilizzo di Angular permette alle applicazioni di essere eseguite interamente dal web browser, previo download da parte del web server. L'elaborazione è dunque interamente *client-side*, e si evidenzia dunque il conseguente risparmio in termini prestazionali poichè non risulta necessario reinviare la pagina web al *web server Apache* implementato, in funzione di ogni richiesta di azione da parte dell'utente. Il codice generato da Angular è correttamente gestito da tutti i principali web browser moderni.
    - Angular inoltre, in combinazione con il toolkit open source Bootstrap, garantisce la full responsivity degli appicativi: il design del sito web, in funzione delle geometrie e delle soluzioni grafiche adottate, si adatta alle dimensioni del dispositivo utilizzato

A partire dai dati immessi in input dagli utenti e/o servizi client, viene quindi generato un JSON.

Il JSON prodotto contiene l'intero flusso informativo oggetto di elaborazione.

Complementarmente, i dati in output offerti dal server per soddisfare le richieste dei client vengono elaborati a partire dai dati forniti dal server.

Il *parser* JAX-RS dedicato si occupa infatti di generare la stringa JSON che verrà resa intellegibile in interfaccia ad uso degli utenti.

Per lo sviluppo di questo strato software è stato utilizzato lo standard/framework JAX-RS con **RESTEasy** come libreria di implementazione.

Il Repository dove consultare tutti i servizi REST è esposto tramite il framework swagger

All'indirizzo:

PROTOCOLLO//IP:PORTA/services/docs

4.1.1.2. Sistemi di controllo degli input

Tutti i dati ricevuti in input sono sottoposti a verifica di sicurezza da parte del sistema; nello specifico, vengono effettuate opportune operazioni di codifica e sanificazione dei dati al fine di evitare potenziali attacchi di tipo XSS – cross site scripting/ SQL Injection; operazioni di successiva decodifica rendono comunque intellegibile l'informazione al momento dell'eventuale nuova presentazione in interfaccia.
I componenti software dedicati alle operazioni di "sanificazione" sono degli EJB Interceptor, questi sono in grado di intercettare qualsiasi request e grazie alle librerie owasp in utilizzo dall'interceptor stesso sono in grado di sanificare i dati di input.
Le tecnologie e/o framework utilizzati per questa attività sono:

- **EJB Interceptor (famiglia EJB 3.1)**
- **ESAPI-2.1**

4.1.2. Business Logic

Lo strato di business logic si occupa di eseguire le operazioni funzionali sottese ai processi implementati e, complementarmente, di gestire la persistenza e la consistenza dei dati utilizzati nei flussi.

Questo strato è preposto alla manipolazione e/o costruzione delle strutture dati di business model,

vengono dunque eseguite tutte le operazioni necessarie a soddisfare i processi di business facenti parte delle funzionalità applicative, di queste operazioni fanno parte naturalmente anche i controlli di business sul dato, nel caso di mancato soddisfacimento del requisito viene "Lanciata" una eccezione di tipo "Business" che comunica all'utente la motivazione dell'interruzione della funzionalità richiesta.

Questo strato assicura anche l'integrità dei dati, attraverso la gestione di opportune transazioni.

Viene gestita inoltre la parte "Autorizzativa", ovvero prima di entrare in ogni metodo di business viene verificato se l'utente che ha richiesto il servizio ha la facoltà per poterlo fare, in caso contrario viene restituito un errore con codice **403 Operazione non consentita**

Le azioni sopra descritte (gestione transazioni, sicurezza) vengono svolte con l'aiuto del framework EJB 3.1, in particolare sono utilizzati i costrutti

- **javax.ejb.TransactionManagement**
- **javax.annotation.security.RolesAllowed**
- **javax.ejb.Stateless**

Lo strato di business logic interagisce con altri strati applicativi in funzione del tipo di funzionalità invocata.
Può ad esempio invocare lo strato "DAO" che si occuperà a sua volta di persistere il dato su base dati oppure interagirà con lo strato di integrationLogic se è necessario invocare un servizio esterno come, ad esempio quello di autenticazione (WSO2).

4.1.3. Persistenza dei dati

Per quanto concerne la persistenza e la consistenza dei dati nei flussi gestiti dai server, l'obiettivo è demandato al framework Hibernate, implementante le Java Persistence API (JPA).

Attraverso il pattern architetturale DAO (Data Access Object), vengono quindi rese disponibili classi (con relativi metodi) che rappresentano ognuna N entità tabellari di database, al fine di stratificare e isolare l'accesso alle tabelle tramite query (poste all'interno dei metodi delle classi), e dunque al layer dati da parte della Business Logic. Questa scelta architetturale crea un elevato livello di astrazione e, di conseguenza, una più facile manutenibilità. I metodi del DAO, con le rispettive query, verranno così richiamati dalle classi della Business Logic, fornendo un mapping (1 DAO a N entità tabellari) delle classi Java nelle tabelle del database relazionale.

Complementarmente, inoltre, la scelta architetturale permette di gestire il reperimento degli oggetti dal database, generando ed eseguendo automaticamente le query SQL necessarie al recupero del flusso informativo e la successiva nuova istanza dell'oggetto precedentemente mappato sul database.

Attraverso la generazione automatica (ove possibile) di query SQL e/o HQL da parte del framework, vengono rese non necessarie implementazioni di recupero e conversione "manuali" dei dati, e viene così garantita la massima portabilità in tutti i database SQL.

4.1.3.1 Funzioni Timer

Al fine di implementare a livello informatico il complesso sistema di vincoli e controlli previsti dalla normativa sottesa ai processi del Fondo, il sistema è fornito di processi ad attivazione temporale.

Le funzioni timer hanno il compito di verificare la corretta esecuzione degli specifici flussi funzionali all'interno della Business Logic.

Una ulteriore implementazione assimilabile alle funzioni citate, è costituita da una set di funzioni atte a sanare eventuali criticità avvenute durante la giornata; si cita in particolare, costituendo un significativo elemento a livello di processo, la funzione di re-invio delle email risultanti in stato KO, e dunque non recapitate agli utenti finali di interesse

La tecnologia con cui sono stati implementati i timer è quella degli EJB timer, in particolare sono utilizzati i seguenti costrutti

- **javax.ejb.TimerService**

tutti i timer del sistema risiedono all'interno del container EJB dell'application server wildfly 13.

Lo strato applicativo dedicato ai timer interagisce, e dunque ha dipendenze, verso gli strati di business logic e integration logic(ove necessario).

Di seguito viene fornito l'elenco dei timer attualmente attivi nel sistema.

- **TIMER_RICHIEDI_AGGIORNAMENTO_APL**
- **TIMER_RIATTIVA_UTENTE_APL**
- **TIMER_PREAVVISO_RINNOVO_FAD**
- **TIMER_RICHIESTA_SCADUTA_FAD**
- **TIMER_RIATTIVA_UTENTE_FAD**
- **TIMER_RINNOVO_SCADUTO_FAD**
- **TIMER_PREAVVISO_RINNOVO_TUTOR**
- **TIMER_RICHIESTA_SCADUTA_TUTOR**
- **TIMER_RIATTIVA_UTENTE_TUTOR**
- **TIMER_PREAVVISO_RINNOVO_DOCENTE**
- **TIMER_RICHIESTA_SCADUTA_DOCENTE**
- **TIMER_RIATTIVA_UTENTE_DOCENTE**
- **TIMER_RINNOVO_SCADUTO_DOCENTE**
- **TIMER_PREAVVISO_RINNOVO_SEDE_OPERATIVA**
- **TIMER_RICHIESTA_SCADUTA_SEDE_OPERATIVA**
- **TIMER_RIATTIVA_UTENTE_SEDE_OPERATIVA**
- **TIMER_RINNOVO_SCADUTO_SEDE_OPERATIVA**
- **TIMER_RIATTIVA_UTENTE_DOCENTE_SINDACALE**
- **TIMER_ANNULLA_PROGETTO_BASE_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_ONTHEJOB_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_PROFESSIONALE_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_QUAPROF_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_RIQPROF_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_QUAPROFAFF_SCADUTO**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_BASE**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_ONTHEJOB**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_PROFESSIONALE**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_QUAPROF**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_RIQPROF**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_QUAPROFAFF**
- **TIMER_NOTIFICA_INTEGRAZIONE_DOMANDA_SAR**
- **TIMER_SCADENZA_RICHIESTE_INTEGRAZIONE_DOMANDA_SAR**
- **TIMER_ATTIVAZIONE_ISTANZA_MOL**
- **TIMER_RIATTIVAZIONE_ISTANZA_MOL_SOSPESA**
- **TIMER_RENDICONTAZIONE_ISTANZA_MOL_VERIFICA_STATO_PROGETTI**
- **TIMER_INVIA_EMAIL_IN_ERRORE**
- **TIMER_CONTROLLI_VERIFICHE_ITINERE**
- **TIMER_RIFIUTO_MANCATA_INTEGRAZIONE**

- **TIMER_SCADENZA_VERIFICA_RENDICONTAZIONE**
- **TIMER_SCADENZA_INVIO_RENDICONTAZIONE**
- **TIMER_SCADENZA_INTEGRAZIONE_VERIFICA_RENDICONTAZIONE**
- **TIMER_ETL_VERIFICA_RENDICONTAZIONE**
- **TIMER_RIPARIZIONE_RISORSE**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_PROFTI**
- **TIMER_RESPINGI_NO_INTEGRAZIONI_PROGETTO_DIRMIR**
- **TIMER_ANNULLA_PROGETTO_PROFTI_SCADUTO**
- **TIMER_ANNULLA_PROGETTO_DIRMIR_SCADUTO**

Integration Logic

Come accennato nel paragrafo precedente lo strato Integration Logic è lo strato applicativo al quale è demandata la preparazione e l'interlocuzione tra i servizi di FTWeb e i servizi esterni utilizzati dal sistema per completare i processi.

In questo strato sono gestite le chiamate dirette ai seguenti sistemi:

- sistema di protocollo/documentale Auriga
    - esposto in modalità SaaS
- sistema di autenticazione WSO2.

Le chiamate vengono effettuate attraverso l'utilizzo del protocollo SOAP *(Simple Object Access Protocol).*

L'utilizzo di SOAP permette dunque lo scambio di pacchetti informativi, in forma di "messaggi", per la gestione di richieste provenienti da sistemi necessitanti interazione a livello di processo.

A livello di struttura, si illustra di seguito una esemplificazione di un messaggio SOAP

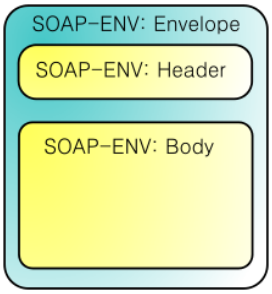


Figura 1 - Messaggio SOAP

Interazione con Auriga

L'interazione con Auriga permette al sistema FTWeb di ottenere nuovi protocolli per documenti da caricarsi ex novo a sistema, o di integrare un protocollo già esistente con ulteriore documentazione.

Lo *stream* dati, opportunamente configurato, permette quindi ai sistemi dialoganti di identificare il documento oggetto di elaborazione e di gestirlo conformemente alle specifiche necessità.

Si riporta di seguito, a questo proposito, un template di messaggio/richiesta inviato dal sistema FTWeb verso Auriga in relazione ad un nuovo allegato da aggiungere ad un protocollo esistente:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Body xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<ns2:service xmlns:ns2="http://updunitadoc.webservices.repository2.auriga.eng.it">
              <codApplicazione> </codApplicazione>
              <istanzaApplicazione> </istanzaApplicazione>
              <userName> </userName>
              <password> </password>
```

```
            <xml>&lt;?xml version="1.0" encoding="UTF-8"
 standalone="yes";UDDaAgg;EstremiXIdentificazioneUD;IdUD;/IdUD;/EstremiXIdentificazioneUD;Nu
 ovoAllegatoUD;VersioneElettronica;NroAttachmentAssociato;/NroAttachmentAssociato;NomeFile&g
 t;/NomeFile;/VersioneElettronica;/NuovoAllegatoUD;/UDDaAgg;</xml>
            <hash></hash>
      </ns2:service>
</SOAP-ENV:Body>
```

4.2.Architettura dei componenti

L'architettura del sistema è tecnicamente suddivisa in moduli software separati, la strategia di separazione è legata all'operatività funzionale dei moduli stessi, ove possibile si è cercato di non creare dipendenze tra i moduli stessi.
Ogni modulo a sua volta è diviso in tre sotto-moduli, ognuno dei quali gestisce autonomamente il tipo di dato consono allo strato applicativo associato.
Ad esempio, il modulo che si occupa delle registrazioni è così strutturato:

- moduloRegistrazione
    - moduloRegistrazione-RS
    - moduloRegistrazione-API
    - moduloRegistrazione-CORE

La parte "RS" contiene tutte le interfacce dei servizi REST contattabili dall'esterno
La parte "API" contiene tutti i modelli dati che vengono esposti dalle interfacce REST
La parte "CORE" contiene i modelli dati di business e tutte le logiche di business relative al modello stesso(in questo caso le sole logiche che riguardano le registrazioni).
Tale stratificazione è applicata a tutti i macro-moduli del sistema.
Di seguito vengono elencati tutti i macro-moduli del sistema.

Modulo common

Rappresenta il modulo che contiene funzioni e operazioni comuni a tutti i moduli, come ad esempio l'invio delle email, l'upload dei file e la gestione delle tabelle tipologiche.

Le operazioni sopra citate infatti sono utilizzate da tutti i contesti applicativi (politiche attive, passive, registrazioni) ad esempio la struttura di una mail (mittente, destinatario, oggetto e body) è la stessa per tutti i contesti, quindi è stata astratta e posizionata in questo modulo.

Per quanto appena descritto questo modulo è indipendente dagli altri, ma gli altri hanno dipendenza verso di lui.

Modulo gestione degli accessi

Rappresenta il modulo che si occupa di tutte le logiche applicative e del modello dati che riguarda la gestione della profilazione degli utenti, non che la creazione di nuove utenze di sistema.

La creazione delle autorizzazioni, dei gruppi, dei nuovi utenti e delle associazioni tra essi è gestita interamente da questo modulo.

Lo strato integration logic di questo modulo interagisce inoltre con il sistema di SSO WSO2 per la registrazione e/o modifica degli utenti sul sistema di SSO stesso.

L'interazione tra il modulo e il SSO avviene tramite un cliente sviluppato esternamente e facente parte della famiglia del framework di sistema EAEAM.

Modulo politiche attive

Rappresenta il modulo che si occupa di tutte le logiche applicative e del modello dati che riguarda la gestione dei piani formativi.

La creazione dei piani formativi e tutte le operazioni annesse come ad esempio presentazione, rendicontazione, variazione etc., sono gestite all'interno di questo modulo.

Lo strato integration logic di questo modulo interagisce inoltre con il sistema AURIGA per la protocollazione dei piani stessi e per la verifica della firma digitale (ove necessario) dei documenti allegati ai piani stessi

L'interazione tra il modulo e Auriga avviene tramite un cliente sviluppato esternamente e facente parte della famiglia del framework di sistema EAEAM.

Modulo politiche passive

Rappresenta il modulo che si occupa di tutte le logiche applicative e del modello dati che riguarda la gestione delle pratiche SAR e MOL.

La creazione delle suddette pratiche e tutte le operazioni annesse come ad esempio la protocollazione, le integrazioni e tutto il flusso di business sono gestite all'interno di questo modulo.

Lo strato integration logic di questo modulo interagisce inoltre con il sistema AURIGA per la protocollazione delle domande e per la verifica della firma digitale (ove necessario) dei documenti allegati alle domande.

L'interazione tra il modulo e Auriga avviene tramite un cliente sviluppato esternamente e facente parte della famiglia del framework di sistema EAEAM.

Modulo registrazioni

Rappresenta il modulo che si occupa di tutte le logiche applicative e del modello dati che riguarda le registrazioni anagrafiche delle entità di sistema.

La creazione delle anagrafiche di APL, SINDACATI, ENTI etc. e tutte le operazioni annesse come ad esempio il flusso autorizzativo da parte del fondo le richieste integrazioni sono gestite all'interno di questo modulo.

Modulo visitare

Rappresenta il modulo che si occupa di tutte le logiche applicative e del modello dati che riguarda l'interazione tra l'app visitare e ftweb.

Questo modulo espone tutti i servizi web utilizzati dall'APP visitare, come ad esempio la visualizzazione delle fasce orarie, l'aggiornamento dello stato delle fascie stesse, la visualizzazione del registro presenze etc.

Modulo Report

Rappresenta il modulo che si occupa della produzione di tutta la reportistica attualmente presente nel sistema.

Racchiude la tecnologia JASPER REPORT e tutte le logiche "tecniche" utili alla produzione dei report stessi.

Tutti i moduli applicativi che hanno necessità di produrre report interagiscono tramite tecnologia EJB con questo modulo.

Modulo Autenticazione

Rappresenta il modulo che si occupa di verificare se gli utenti che effettuano le richieste siano già autenticati nel sistema e in caso contrario redirige l'utente stesso verso il sistema SSO per consentirgli di effettuare l'autenticazione che quindi viene fatta esternamente a ftweb.

Questo modulo è sviluppato tramite un "Servlet Filter" che per sua natura intercetta qualunque richiesta viene fatta al sistema, questo comportamento ci consente di applicare tutte le logiche necessarie a capire se il tipo di richiesta necessita obbligatoriamente di un utente autenticato.

Dopo che l'utente effettua l'autenticazione su WSO2, WSO2 stesso redirige l'utente nuovamente verso ftweb, il modulo tramite la http response proveniente da WSO2 controlla nuovamente se l'utente è autenticato, in caso positivo, tramite il modulo di gestione degli accessi sopra descritto, viene verificato se l'utente è censito all'interno di ftweb (l'utente potrebbe essere utente di SSO perché censito su altri applicativi, ma non di ftweb) e in caso positivo vengono recuperate tutte le autorizzazioni associate all'utente, tali autorizzazioni vengono censite all'interno dell'EJB container per tutta la durata della sessione utente.

Questo meccanismo consente in maniera automatica e senza scrittura di procedure "lato codice" di verificare se oltre al fatto che l'utente è autenticato è anche autorizzato ad effettuare una determinata operazione.

Il seguente schema logico che sintetizza e mostra le interazioni tra ftweb e il SSO