Fascicolo Sanitario Elettronico di seconda generazione: nodi aggregati specializzati e sistema di autorizzazione/autenticazione

Technical Report · November 2014

DOI: 10.13140/RG.2.1.1827.9929

4 authors:

Antonino Fiannaca

Italian National Research Council

28 PUBLICATIONS **79** CITATIONS

SEE PROFILE

Antonio Messina

Italian National Research Council

17 PUBLICATIONS **13** CITATIONS

SEE PROFILE

Pietro Storniolo

Italian National Research Council

27 PUBLICATIONS **70** CITATIONS

SEE PROFILE

Alfonso Urso

Italian National Research Council

60 PUBLICATIONS **206** CITATIONS

SEE PROFILE

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Fascicolo Sanitario Elettronico di seconda generazione: nodi aggregati specializzati e sistema di autorizzazione/autenticazione

A. Fiannaca, A. Messina, P. Storniolo, A. Urso

Rapporto Tecnico N.:
RT-ICAR-PA-14-04 **Novembre 2014**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sede di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sede di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Fascicolo Sanitario Elettronico di seconda generazione: nodi aggregati specializzati e sistema di autorizzazione/autenticazione

A. Fiannaca[1], A. Messina[1], P. Storniolo[1], A. Urso[1]

Rapporto Tecnico N.:
RT-ICAR-PA-14-04 **Novembre 2014**

---

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sede di Palermo, Viale delle Scienze edificio 11, 90128 Palermo.

Sommario

1 Introduzione

Secondo quanto definito dalle linee guida nazionali del Ministero della Salute, il Fascicolo Sanitario Elettronico è composto dall'insieme di dati e documenti digitali di tipo sanitario e socio-sanitario generati da eventi clinici presenti e trascorsi, riguardanti l'assistito. La gestione di queste informazioni sul territorio nazionale ha richiesto l'adozione di un modello in grado di affrontare innanzitutto il problema dell'interoperabilità (o compatibilità) tra le soluzioni esistenti e già applicate nelle singole regioni. E' necessario che il modello adottato, sia in grado di gestire i tre livelli di interoperabilità: strutturale, semantica e dei servizi.

- **Interoperabilità Strutturale**. Essa attiene alla possibilità che diversi sistemi informativi ospedalieri siano in grado di cooperare tra di loro per la memorizzazione e la consultazione dei documenti sociosanitari, attraverso un sistema che distribuito che, dal punto di vista logico, si pone come un sistema informativo unico. In questa ottica è necessario garantire l'interazione tra diversi modelli di indicizzazione dei documenti, tra modalità e policy di accesso differenti e tra strutture di rappresentazione dei dati diverse, in maniera coerente, mascherando tutte le disomogeneità presenti all'utente finale. Ciò può avvenire o mediante l'adozione di standard condivisi, o attraverso la presenza di un opportuno middleware al quale interfacciare i singoli sistemi. Per quanto concerne le strutture dati complesse, come ad esempio l'e-prescription ed il Patient Summary, si tende a garantire l'interoperabilità attraverso l'adozione dello standard HL7 (Health Level 7), con particolare riferimento al CDA (Clinical Document Architecture) release 2.

- **Interoperabilità Semantica.** Essa riguarda la necessità di attribuire lo stesso significato allo stesso termine, così come stabilire che termini diversi sono semanticamente equivalenti. Ciò rappresenta indubbiamente un fattore chiave per garantire una corretta gestione dei percorsi assistenziali e delle procedure amministrative ad essi connessi. L'adozione di sistemi di codifica condivisi (come LOINC, ICD e SNOMED) e la standardizzazione della terminologia costituiscono le principali azioni al fine di garantire l'interoperabilità semantica nei molteplici contesti applicativi. Inoltre, si noti che l'adozione dello standard HL7-CDA2 garantisce anche l'interoperabilità semantica dei documenti clinici.

- **Interoperabilità dei Servizi.** Si riferisce alla cooperazione di più servizi all'interno dei processi sanitari. Un esempio di cooperazione tra servizi sviluppati indipendentemente è rappresentato dal processo che prevede la prenotazione di una prestazione mediante CUP e quello che gestisce il pagamento del relativo ticket. Ciò significa che diversi sistemi erogatori di servizi devono poter interoperare in maniera trasparente verso l'utente (o verso

un altro erogatore). In particolare, il Sistema Pubblico di Cooperazione (SPCoop) fornisce alcuni strumenti per la cooperazione tra servizi erogati dalla Pubblica Amministrazione.

Il progetto congiunto Infrastruttura tecnologica del Fascicolo Sanitario Elettronico (InFSE) ha avuto, tra gli altri obbiettivi, anche quello di elaborare un insieme di linee guida di riferimento in grado di affrontare e superare il problema dell'interoperabilità di FSE, implementando un'infrastruttura tecnologica del Fascicolo Sanitario Elettronico (FSE) del cittadino, condivisa a livello nazionale ed allineata allo scenario europeo.

2 L’erogazione di nuovi servizi nel FSE2.0

2.1 Introduzione

Il Fascicolo Sanitario Elettronico di prima generazione ha come obiettivo il fornire agli operatori del Sistema Sanitario una visione globale e unificata dello stato di salute dei singoli cittadini. La tipologia di informazioni che possono essere aggregate e condivise si limita, quindi, ad una collezione di eventi sanitari e documenti di sintesi che contengono la storia clinica del singolo cittadino, organizzati secondo una struttura gerarchica paziente-centrica, che permette la navigazione fra i documenti clinici in modalità differenti a seconda del tipo di indagine (sia essa originata dal cittadino stesso o da un operatore del Sistema Sanitario). Per tale ragione, tutte le interrogazioni a cui è in grado di rispondere il FSE di prima generazione ricadono sempre e solo sullo stato clinico del singolo cittadino, seppur con un potenziale orizzonte temporale di riferimento che è la sua intera vita.

Al fine di favorire la qualità, il monitoraggio, le statistiche di tipo epidemiologico, nonché l'appropriatezza nella dispensazione dei farmaci e dei presidi sanitari, così come i riscontri delle terapie per singole patologie, il Fascicolo Sanitario Elettronico di seconda generazione introdurrà dei componenti aggiuntivi che ne estenderanno le funzionalità.

2.2 I nodi InFSE

Il modello architetturale di InFSE si compone di una infrastruttura di nodi (locali e regionali) che hanno, tra le altre cose, il compito di immagazzinare tutte le informazioni relative agli eventi clinici registrati per ogni singolo cittadino, che verranno archiviati e gestiti dal componente “Gestore dei Documenti”. Le principali tipologie di informazione che il Fascicolo Sanitario Elettronico raccoglie e rende disponibili al medico e al paziente, sono riportate in figura 1.

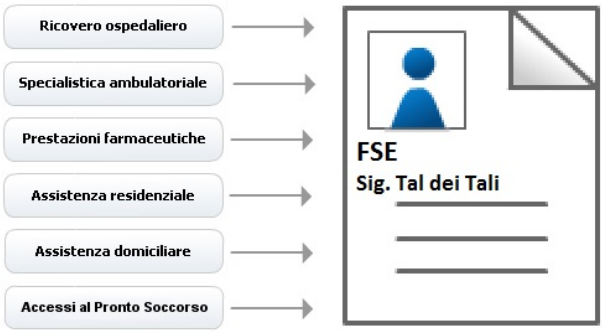


Figura 1 - Informazioni che il FSE di prima generazione può ad immagazzinare

Allo stato attuale, quindi, una possibile soluzione per estendere le funzionalità del FSE alle interrogazioni che riguardano il singolo dato (anziché il singolo paziente), sarebbe quella di generare una richiesta che controlli tutti gli eventi clinici registrati da tutti i cittadini che hanno autorizzato la visione di quel determinato dato clinico. Ad esempio, se fosse necessario effettuare il monitoraggio periodico della pressione sistolica arteriosa registrata in tutti pazienti ricoverati presso le strutture sanitarie perché affetti da ipertensione, sarebbe necessario realizzare una query federata (secondo le specifiche InFSE) su tutti i documenti clinici presenti per ogni paziente conservati nei nodi di riferimento.

Chiaramente tale soluzione, non è praticabile per differenti ragioni, come ad esempio i tempi di attesa della ricerca o il volume di traffico generato. E' possibile allora ricorrere a soluzioni alternative, come l'introduzione di nuovi nodi che si occupano ad esempio di aggregare eventi sanitari o di gestire informazioni non provenienti da operatori del Sistema Sanitario.

2.3 Nodi speciali in InFSE 2.0: nodo aggregato e nodo taccuino

Al fine di immagazzinare e rendere fruibili delle informazioni "trasversali" rispetto a quanto previsto dal FSE di prima generazione, è necessario organizzare una raccolta di eventi caratteristici, indicizzati in maniera parallela a quanto offerto dal FSE stesso, in grado di sfruttare tutte le funzionalità messe a disposizione dall'infrastruttura InFSE.

In particolare, una possibile soluzione è rappresentata dalla realizzazione all'interno della infrastruttura InFSE, di una collezione di nodi specializzati per una particolare ricerca trasversale (ossia orientata non al paziente, ma allo studio di uno specifico insieme di dati clinici). Tali nodi prendono il nome di **nodi aggregati**, in quanto hanno la stessa struttura dei nodi InFSE, ma contengono esclusivamente una aggregazione di eventi che vengono registrati unicamente qualora, durante l'inserimento di un evento clinico, siano soddisfatte delle condizioni predeterminate.

Alcune tra le caratteristiche fondamentali dei nodi aggregati sono elencate di seguito:

- Il nodo aggregato è un nodo passivo, in quando viene popolato tramite ricezione di eventi provenienti da altri nodi. I dati in esso contenuti non sono direttamente modificabili né dal cittadino, né dagli operatori della sanità;
- Il nodo aggregato raccoglie informazioni previa accettazione del cittadino, il quale potrà firmare un opportuno consenso al trattamento dei dati personali legati a studi sperimentali (tipicamente indagini epidemiologiche), per ogni singola patologia, in aggiunta a quanto previsto per l'utilizzo dalle funzionalità standard di FSE;

- A differenza dei nodi InFSE standard, per questa tipologia di nodo non è vantaggiosa una gerarchia ramificata nel territorio (nodi locali e regionali), ma piuttosto una centralizzazione quantomeno a livello regionale;

In figura 2 è riportato un esempio di interazione tra due nodi InFSE standard ed un nodo aggregato che raccoglie eventi relativi alla patologia ipertensione. È possibile notare come soltanto le diagnosi di tipo "ipertensione" fanno scaturire un evento che popola il registro del nodo aggregato. Inoltre, come accennato prima, il nodo aggregato "ipertensione" è presente con una sola istanza a livello regionale, in quanto è ipotizzabile che nodi di questo tipo siano presenti in forma ridotta sul territorio nazionale per ottimizzare le ricerche e gli studi sui pazienti affetti dalle diverse patologie.

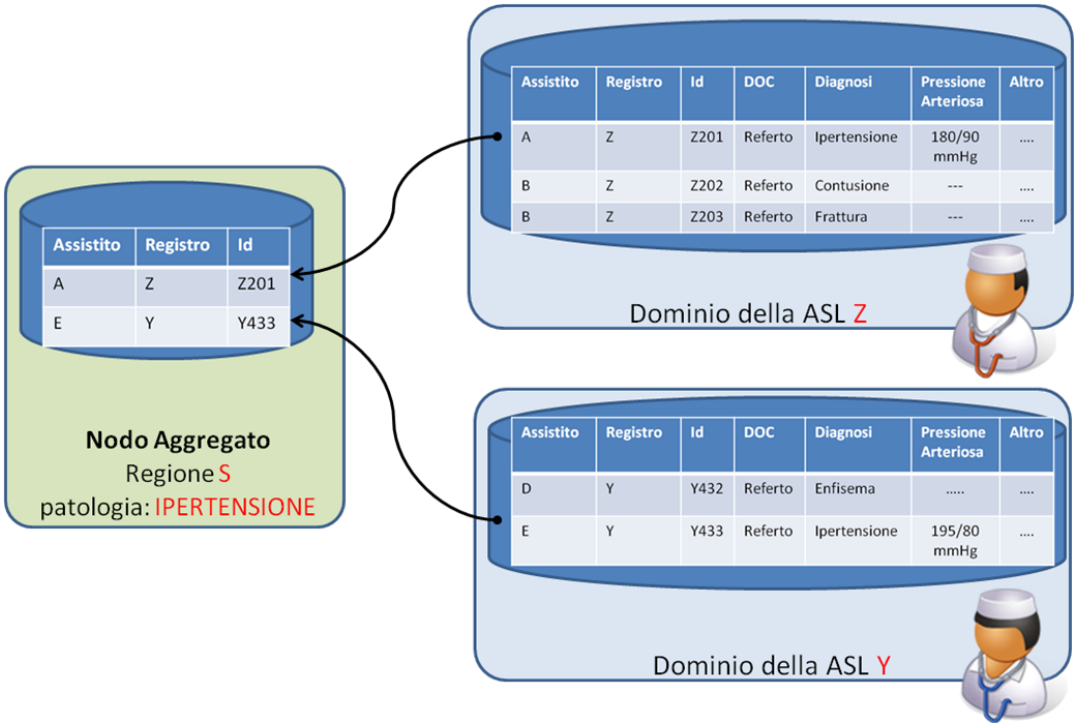


Figura 2 - Esempio di interazione tra nodi InFse standard e nodi aggregati

Un'altra categoria di nodi che verrà introdotta nel FSE2.0 al fine di estendere le funzionalità di InFSE al taccuino del cittadino è quella relativa al **nodo taccuino.** Secondo quanto previsto, infatti, il fascicolo di seconda generazione dovrà mettere a disposizione del cittadino una sezione riservata a (1) dati e informazioni personali, (2) file di documenti sanitari, (3) un diario degli eventi rilevanti e (4) un promemoria per i controlli medici periodici, arricchendo il FSE con informazioni a supporto della descrizione dello stato di salute. Ovviamente tali informazioni dovranno essere distinte da quelle inserite dal personale della sanità, principalmente per il fatto che il taccuino rappresenta una collezione di informazioni non validate dal sistema sanitario. Un'altra caratteristica che dovrà rispettare il nodo taccuino, sarà quella di raccogliere

informazioni per lo più non strutturate, come appunto le annotazioni personali che il cittadino ritiene di dover conservare come informazione utile del suo stato di salute.

A differenza di tutti gli altri nodi, i nodi taccuino offrono al cittadino la possibilità di interagire con le informazioni raccolte sia in lettura che in scrittura, consentendo l'immissione di nuove annotazioni, così come la modifica degli stati di salute attuali o pregressi. Un esempio di interazione col nodo taccuino è riportato in figura 3.

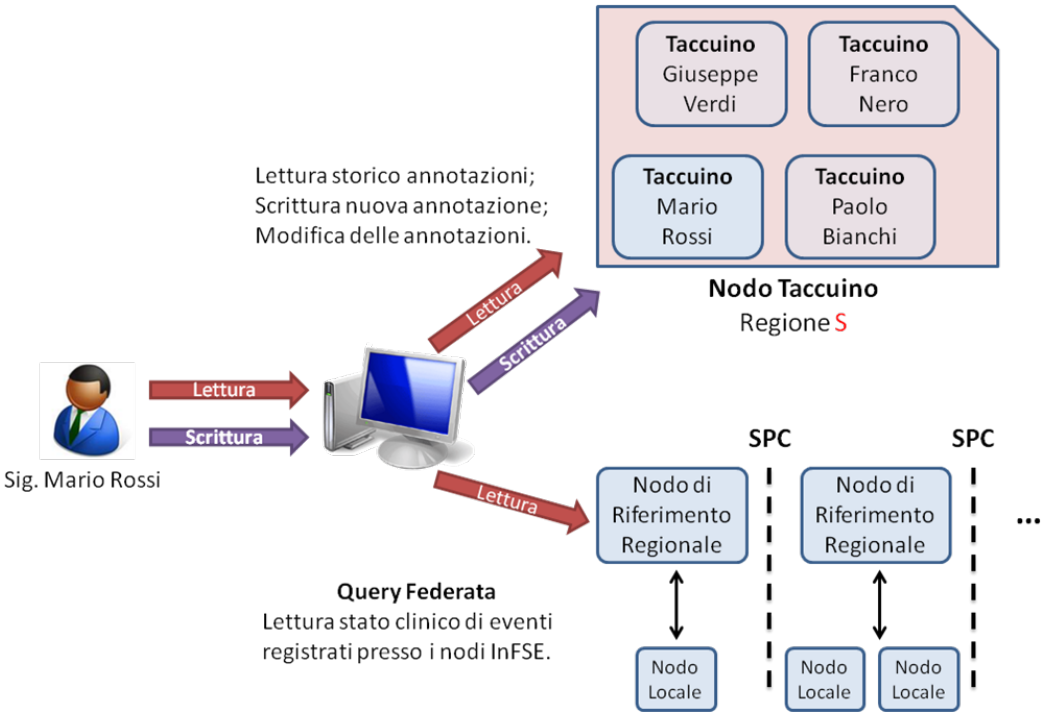


Figura 3 - Esempio di interazione del cittadino con nodo taccuino e nodi FSE standard

2.4 Scenari Nodi Aggregati

2.4.1 Registrazione dell'Assistito 123 al nodo aggregato Patologia N.

Supponiamo che il cittadino "Assistito 123" dia il consenso affinché il nodo aggregato possa raccogliere gli eventi relativi alla Patologia N.

In questo scenario, l'attore principale dovrà accedere al sistema tramite il portale web della Regione S di appartenenza e dare l consenso affinché avvenga la registrazione al nodo aggregato in questione.

L'accesso al nodo avverrà attraverso l'infrastruttura InFSE, che si occuperà di abilitare la registrazione degli eventi sanitari relativi alla Patologia N, presso il nodo

aggregato corretto. In Figura 4 è rappresentato il percorso della richiesta, dal portale web fino al registro, seguendo il percorso delle frecce rosse.

In seguito alla registrazione dell'Assistito 123, ogniqualvolta si verificherà l'evento sottoscritto, esso verrà indicizzato nel registro del nodo Aggregato Patologia N.

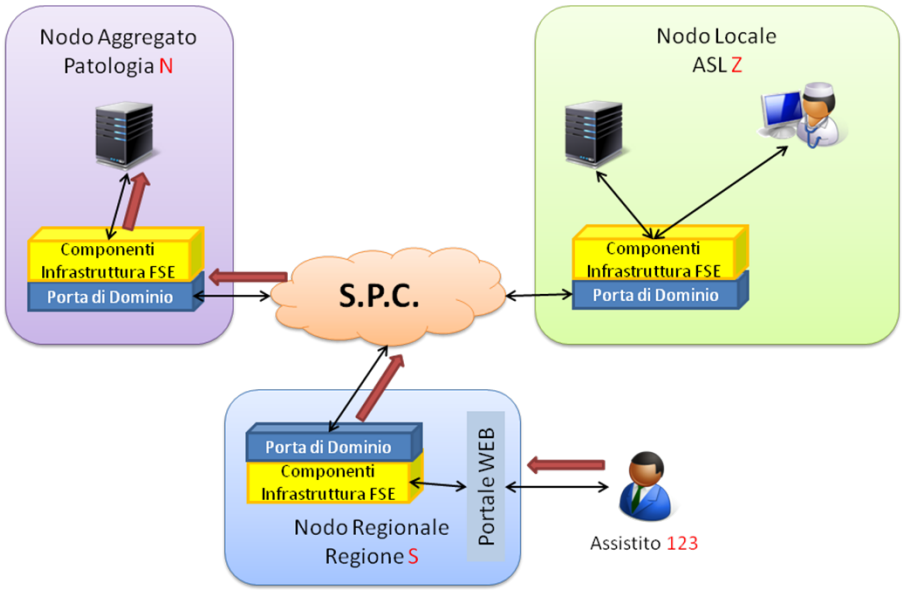


Figura 4 - Registrazione dell'Assistito 123 al nodo aggregato Patologia N

2.4.2 Evento sanitario dell'Assistito 123 presso il nodo locale ASL Z relativo alla Patologia N. Interrogazione Nodo Aggregato per verificare registrazione Assistito 123.

Supponiamo che il cittadino "Assistito 123" si rechi presso la ASL Z, che rappresenta un nodo locale completo InFSE.
Supponiamo che la ASL Z abbia aderito alla sub-federazione di nodi che consente la condivisione dei dati registrati presso l'operatore sanitario, con i nodi aggregati.
Supponiamo che egli si sottoponga ad una analisi clinica relativa alla Patologia N.
Nel momento in cui vengono inseriti nel SIO i risultati delle analisi, verrà generato un evento che interrogherà l'infrastruttura per sapere se l'assistito è registrato al nodo aggregato Patologia N, come in Figura 5.
Poiché il cittadino si era preventivamente registrato, è autorizzata l'indicizzazione dell'evento sanitario presso il nodo aggregato di riferimento.

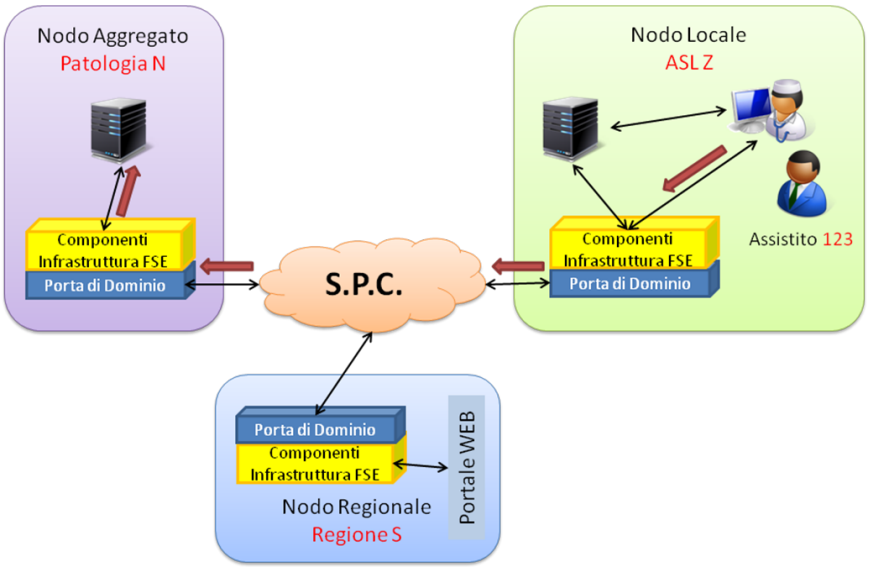


Figura 4 - Evento sanitario dell'Assistito 123 presso il nodo locale ASL Z

2.4.3 Evento sanitario dell'Assistito 123 presso il nodo locale ASL Z relativo alla Patologia N. Indicizzazione nel Nodo Aggregato dell'evento dell'Assistito 123

Supponendo che la registrazione dell'evento presso il nodo aggregato Patologia N sia autorizzato, i dati clinici verranno conservati automaticamente presso il repository del nodo locale ASL Z (Figura6, freccia blu), mentre l'indicizzazione degli stessi dati verrà conservata, attraverso InFSE, presso il nodo aggregato Patologia N (Figura 5, freccia verde). In questo modo, il dato clinico verrà conservato in una repository che non fa parte di InFSE, ma che sarà collegata all'infrastruttura stessa per mezzo dell'indicizzazione presso il nodo aggregato di competenza.

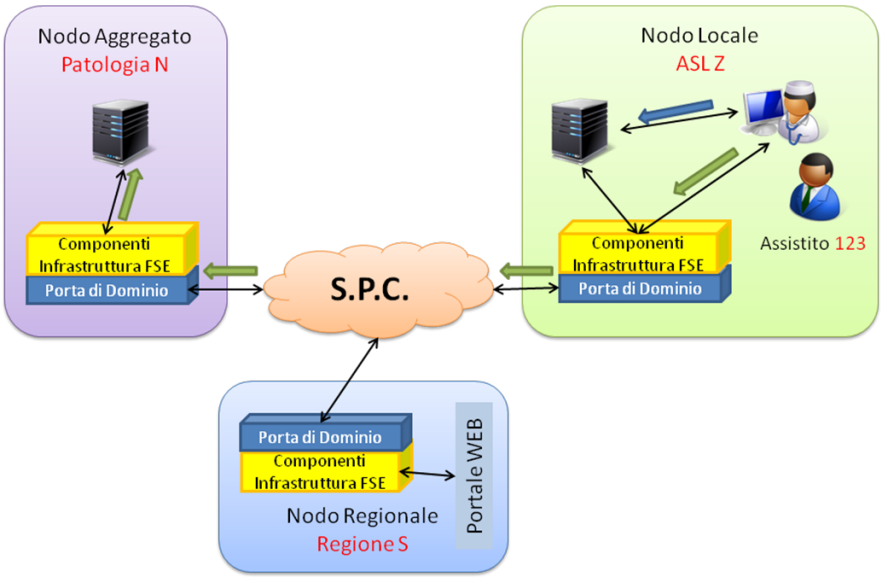


Figura 6 - Indicizzazione nel Nodo Aggregato dell'evento dell'Assistito 123

2.4.4 Interrogazione Nodo Aggregato Patologia N per recupero informazioni

Supponendo che un assistente sanitario voglia visionare tutti gli eventi indicizzati nel nodo aggregato Patologia N, egli dovrà interrogare quest'ultimo attraverso InFSE. Poiché il registro del nodo aggregato contiene esclusivamente i riferimenti ai dati contenuti presso gli storage di riferimento dei nodi locali, esso si comporterà come il nodo di un sistema informativo distribuito che, sfruttando una infrastruttura parallela a quella di InFSE, è in grado di recuperare le informazioni di dettaglio dai nodi locali che hanno scelto di interfacciarsi con tale infrastruttura di condivisione dati. In Figura 7 è evidenziata l'infrastruttura utilizzata per lo scambio dati di dettaglio (frecce blu) tra gli operatori sanitari ASL X, ASL Y e ASL Z (che oltre ad essere federati ad InFSE hanno aderito allo federazione dei nodi aggregati) ed il nodo aggregato per la patologia N.

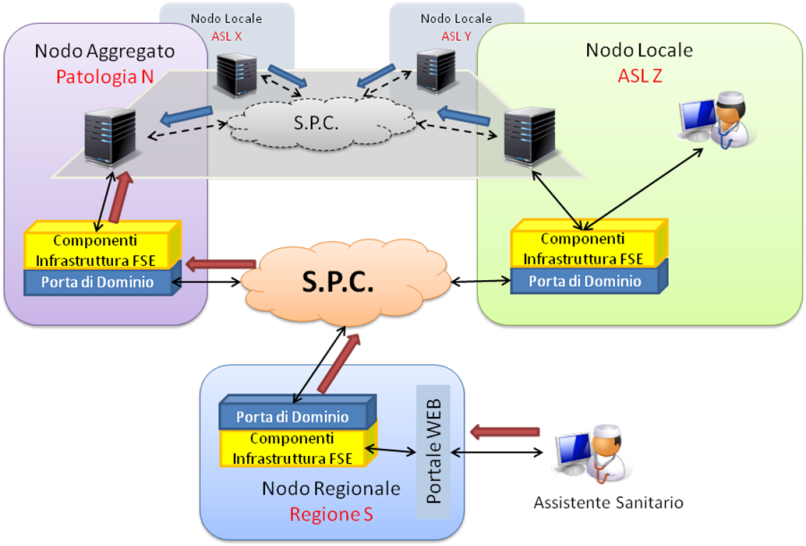


Figura 7 - Interrogazione Nodo Aggregato Patologia N per recupero informazioni

2.5 Scenari nodi Taccuino

2.5.1 Scrittura dell'Assistito 123 sul Taccuino della Regione S di riferimento

Supponiamo che il cittadino "Assistito 123" sia residente presso la Regione S, ed acceda al portale web della regione di appartenenza.

Supponiamo che l'assistito dia il consenso affinché venga attivato il taccuino personale presso il nodo taccuino dell'infrastruttura InFSE.

In questo scenario, il cittadino potrà annotare una serie di informazioni non certificate, ma rilevanti per il suo stato di salute, all'interno del nodo taccuino che comunica direttamente con il nodo regionale. Come evidenziato dalla Figura 8, infatti, il nodo

taccuino è connesso tramite componenti InFSE direttamente al nodo regionale di riferimento, alla stregua di un nodo locale assistito. Pertanto, in questo scenario, le informazioni non certificate inserite dall'assistito non vengono veicolate attraverso il SPC, ma vengono gestite da InFSE direttamente a livello di nodo regionale di riferimento.

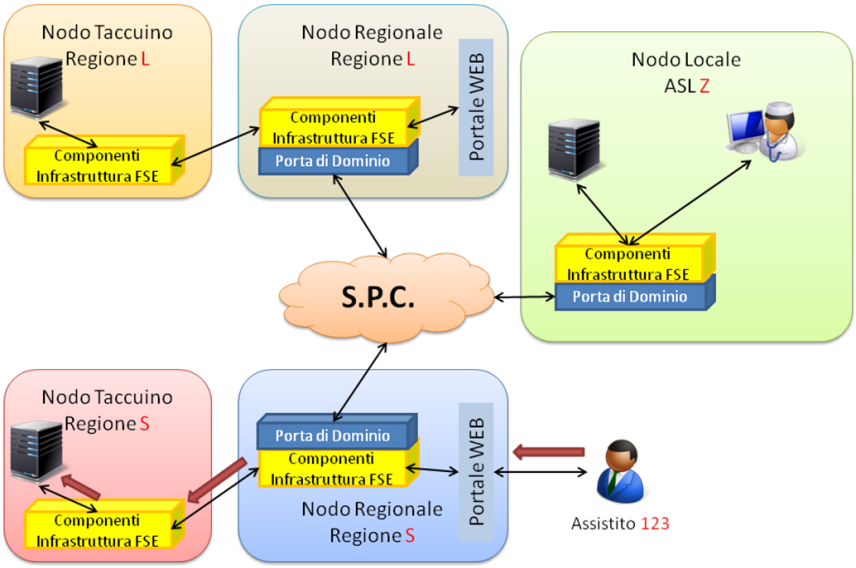


Figura 8 - Scrittura dell'Assistito 123 sul Taccuino della Regione S di riferimento

2.5.2 Lettura Taccuino dell'Assistito 123 attraverso i nodi regionali di riferimento

Supponendo che un assistente sanitario abbia la necessità di visionare il taccuino dell'Assistito 123, egli dovrà effettuare una richiesta federata ad InFSE, che si occuperà di recuperare tali informazioni.

La richiesta federata coinvolgerà i nodi regionali, alla ricerca del taccuino relativo all'Assistito 123; infatti, nel caso in cui l'assistito sia stato residente in diverse regioni, il taccuino sarà formato dalla somma dei taccuini. In Figura 9, l'assistito è stato residente sia in Regione S che in Regione L, ed in entrambe le regioni ha dato il consenso alla visualizzazione del taccuino da parte degli operatori autorizzati. Per tale ragione, la richiesta della ASL Z verrà inoltrata da InFSE ai nodi regionali di riferimento, recuperando così due differenti taccuini per lo stesso assistito.

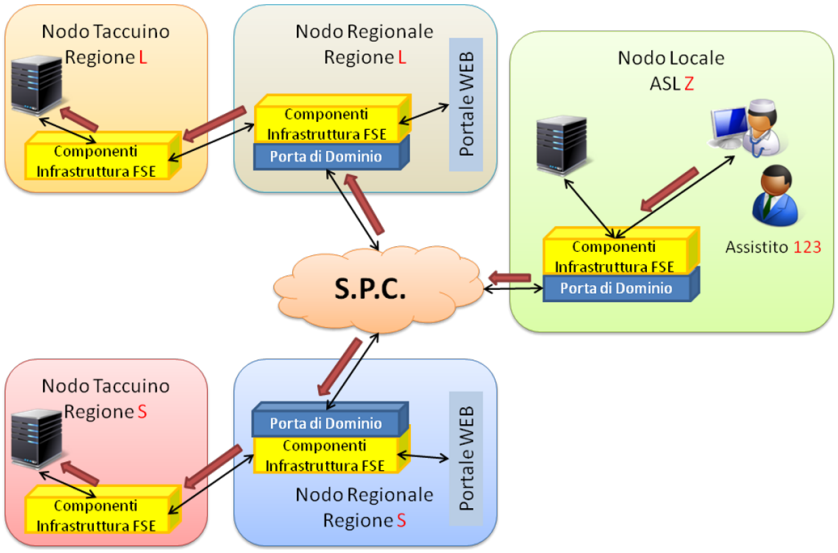


Figura 9 - Lettura Taccuino dell'assistito attraverso i nodi regionali di riferimento

3 Requisiti funzionali dell'Infrastruttura InFSE

3.1 Introduzione

Il modello architetturale deve consentire a tutti gli attori del Servizio Sanitario Nazionale autorizzati di accedere ai documenti sanitari di loro competenza e di gestire l'evoluzione dello stato dei processi sanitari nel tempo.

3.2 Modello architetturale

L'infrastruttura InFSE si è posta, tra l'altro, l'obiettivo di compatibilità con le soluzioni architetturali regionali precedentemente sviluppate, nell'ottica di un modello di infrastruttura federata, condivisa a livello nazionale e allineata allo scenario internazionale. A tale proposito, il ricorso a meccanismi di federazione e a standard e tecnologie aperte e internazionalmente accettate risulta indispensabile. Il modello architetturale deve pertanto essere rispondente a specifiche esigenze progettuali, le più importanti delle quali sono elencate di seguito.

1. Consentire la localizzazione e la reperibilità delle informazioni sanitarie degli assistiti, da parte degli utenti autorizzati.
2. Supportare adeguatamente i processi sanitari, notificando opportuni eventi clinici ad ogni loro occorrenza.
3. Supportare la natura federata e decentralizzata del SSN, al fine di rispettare l'autonomia delle Regioni.
4. Consentire una facile integrazione con sistemi e infrastrutture preesistenti di sanità elettronica.
5. Essere basato su standard aperti e sulle tecnologie cloud, per facilitare l'interoperabilità delle infrastrutture e dei sistemi regionali e minimizzare gli investimenti.
6. Presentare caratteristiche di scalabilità e modularità, allo scopo di consentirne uno sviluppo incrementale e distribuito.
7. Fornire caratteristiche di affidabilità, che la rendano fault-tolerant e priva di single-point-of-failure.
8. Fornire adeguate caratteristiche prestazionali in termini di accessibilità ai documenti, ai dati sanitari ed alle ricerche di tipo statistiche.
9. Garantire un elevato livello di sicurezza nella gestione degli aspetti legati ad identificazione ed autenticazione degli utenti e delle componenti e di accesso ai documenti.
10. Essere basato sul Sistema Pubblico di Connettività (SPC), al fine di rispettare le regole di cooperazione applicativa previste.

11. Essere conforme alle indicazioni del Garante della Privacy per quanto riguarda le normative vigenti in materia di riservatezza e accesso ai dati contenuti nel Fascicolo Sanitario Elettronico.

4 L'infrastruttura per la sicurezza del Fascicolo 2.0

4.1 Introduzione

Le attività di sviluppo nell'ambito della realizzazione del Fascicolo Sanitario Elettronico di Seconda Generazione (FSE2.0) hanno evidenziato tutta una serie di peculiarità ed idiosincrasie dovute alle modalità tipicamente utilizzate nelle interazioni con i registri ebXML.

4.2 InFSE

L'infrastruttura del FSE di seconda generazione si sviluppa per mezzo di apposite componenti in grado di erogare i servizi necessari al funzionamento di quanto precedentemente descritto. Lo strato infrastrutturale si basa sulla piattaforma InFSE che è in grado sia di garantire l'interoperabilità tra le varie soluzioni di fascicolo sanitario regionale, che offrire le caratteristiche di modularità e scalabilità necessarie per ottemperare a quanto richiesto dal fascicolo di seconda generazione.

In particolare, al fine di garantire la realizzazione dell'architettura rappresentata in Figura 1, di cui l'infrastruttura assume un ruolo centrale, è stato fatto un lavoro di adattamento delle componenti InFSE, consentendone l'interfacciamento con i servizi di erogazione previsti, quali appunto l'Adapter verso le appllicazioni locali, il sistema di gestione dell'interazione dei SIO ed infine il portale FSE.

Nell'ottica di fornire un adeguato livello di sicurezza e tenendo in considerazione anche le politiche di consenso all'alimentazione e consultazione del fascicolo, nel rispetto di quanto previsto dalle più recenti linee guida in materia di FSE, è stato curato con particolare attenzione l'aspetto riguardante la gestione delle politiche di accesso, attraverso le varie piattaforme abilitate allo scambio di dati e documenti per mezzo dell'infrastruttura. Più in dettaglio, sono state implementate alcune componenti che si occupano di proteggere le funzionalità offerte dall'infrastruttura (come il gestore dei documenti ed il registro indice federato) attraverso uno strato specifico di sicurezza, che a sua volta si compone di tre macro-componenti, rispettivamente per le funzioni di autenticazione, identificazione ed autorizzazione. Il processo di autenticazione ha come scopo quello di determinare un cosiddetto *trusted context environment* fra l'applicazione richiedente il servizio (fruitore) e l'infrastruttura stessa (erogatore). Tale fase si realizza attraverso l'invio di credenziali che rappresentano l'identità da parte del fruitore e da parte dell'erogatore in modo che entrambi possano attestare la corrispondente attendibilità. A questa segue la fase di identificazione, che ha lo scopo di determinare l'identità del fruitore (utente) assegnandogli eventualmente proporre tutte e sole le operazioni consentite, tramite il meccanismo di assegnazione

di ruoli. Le informazioni necessarie per l'identificazione, ovvero il set di informazioni utili per identificare un soggetto sulla base delle credenziali fornite, vengono indicate nella descrizione del servizio esposto dall'erogatore. L'ultima fase è rappresentata dall'autorizzazione, che consiste nel processo decisionale tramite il quale vengono verificati i diritti dei soggetti che richiedono l'accesso alle varie componenti dell'Infrastruttura FSE per reperire o modificare i dati memorizzati. Il fruitore accede a questa fase soltanto dopo che la sua identità è stata verificata ed è stato costruito il portafoglio di asserzioni, che servono a verificare se il soggetto richiedente ha effettivamente il diritto di compiere il tipo di accesso richiesto; ad esempio, considerando il caso d'uso di acquisizione di documenti clinici, la fase di autorizzazione deve essere eseguita per ogni richiesta ricevuta dal Registro Regionale, perché anche i metadati sono da considerare informazioni sensibili, e devono essere acceduti solo dai soggetti aventi diritto. Il processo di autorizzazione utilizza le informazioni che sono presenti nella richiesta ricevuta dal soggetto richiedente, verificandone le credenziali che ne attestano il ruolo ed i diritti di accesso ai servizi secondo il modello RBAC (Role-Based Access Control).

Dal punto di vista implementativo, per il processo di autenticazione e identificazione, l'erogatore di servizi InFSE (il *Service Provider*) veicola la richiesta del fruitore verso una coppia di entità: l'*Identity Provider* (IdP) e il *Profile Authority* (PA). La prima entità è stata implementata tramite la soluzione di accesso Single Sign-On implementata da Shibboleth, che fornisce un sistema federato di autenticazione ed autorizzazione basato su asserzioni Security Assertion Markup Language (SAML). Tali asserzioni vengono prodotte interrogando la seconda entità, il PA, che si fa carico della gestione e manutenzione dei profili utente, attraverso la produzione di n-ple strutturate valore-attributo. Tale entità sfrutta il protocollo LDAP per la modifica e l'interrogazione dei servizi di un directory database contenente le informazioni sugli operatori, gli assistiti e le loro correlazioni. Chiaramente, al fine di mantenere aggiornate le suddette informazioni, è presente un meccanismo di sincronizzazione tra il PA e l'anagrafica regionale che censisce assistiti ed operatori sanitari MMG e PLS.

La lista degli attributi utilizzati durante la fase di identificazione è la seguente:

- *uid*: Codice Fiscale;
- *isMemberOf*: ruolo dell'utente (e la sua gerarchia secondo quanto stabilito da FSE);
- *seeAlso* (opzionale): lista dei cittadini di cui il presente è tutore legale del fascicolo;
- *manager* (opzionale): cittadino che rappresenta il tutore legale
- *consensoAlimentazione*: può assumere i due valori "TRUE" e "FALSE";
- *consensoConsultazione*: contenente una eventuale lista di ruoli.

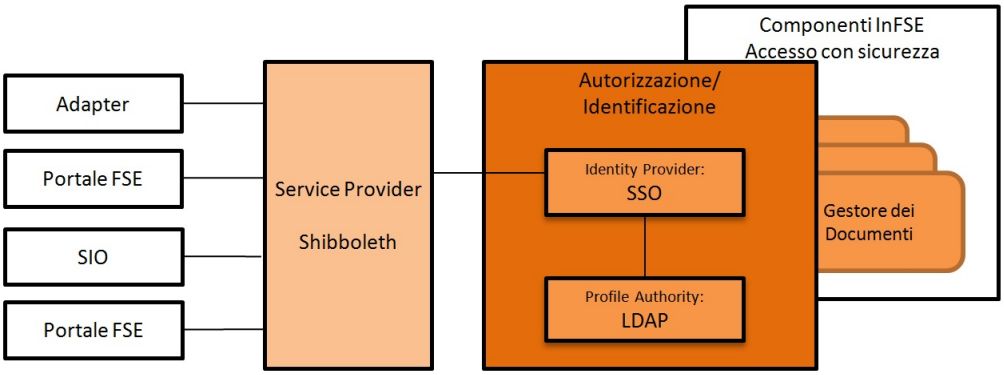


Figura 10 - Schema DI autorizzazione/identificazione infrastruttura FSE

La figura 10 mostra una schematizzazione delle componenti che fanno parte del processo di autenticazione/identificazione all'infrastruttura FSE. In figura è possibile anche vedere come questo meccanismo di identificazione si espone ai fruitori dell'infrastruttura di fascicolo attraverso un Service Provider Shibboleth.

Dopo aver effettuato il processo sopra indicato sarà possibile inviare richieste sicure accompagnate dagli attributi ottenuti nella fase precedente. Per il processo di autorizzazione si rende necessario ricorrere alle due componenti *Policy Enforcement Point* (PEP) e *Policy Decision Point* (PDP), che si occupano rispettivamente di (1) "intercettare" le richieste di accesso rilevanti dal punto di vista della sicurezza verificando la veridicità delle credenziali e della risorsa a cui il PEP è associato e (2) eseguire il processo decisionale per valutare se un dato soggetto ha il diritto di compiere una operazione di accesso ad una risorsa.

Tramite analogo meccanismo di richiesta corredata da portafoglio di asserzioni, l'infrastruttura si occupa anche di proteggere le informazioni presenti sull'anagrafe regionale, qualora un soggetto autorizzato (ad esempio un MMG) avesse necessità di eseguire una interrogazione di dettagli su una lista di assistito.

4.3 Panoramica del protocollo LDAP

Il client inizia una sessione LDAP collegandosi ad un server LDAP (chiamato anche DSA, *Directory System Agent*). Sono comunemente definite due porte TCP per la connessione in chiaro (porta 389) e la connessione cifrata (porta 636). Le comunicazioni sono sempre iniziate dal client che invia una richiesta alla quale il server deve rispondere. Vi sono alcune eccezioni a questo pattern di comunicazione, definite nelle RFC). Tutte le informazioni sono codificate e trasmesse utilizzando le *Basic Encoding Rules* (BER).

Il client può richiedere le seguenti operazioni:

- *Bind*: esegue l'autenticazione
- *Search*: esegue una ricerca

- *Compare*: esegue un test di confronto tra un valore ed il valore assegnato ad un attributo
- *Add*: aggiunge un nuovo oggetto
- *Delete*: cancella un oggetto
- *Modify*: modifica gli attributi di un oggetto
- *Modify Distinguished Name* (DN): sposta o rinomina un oggetto
- *Abandon*: annulla una richiesta inviata in precedenza
- *Extended Operation*: richiesta di operazioni estese (definite in altre RFC)
- *Unbind*: indica al server di chiudere la connessione (non è esattamente l'inverso della Bind)
- *StartTLS*: estensione per utilizzare Transport Layer Security (TLS) per eseguire la Bind

Il server può inviare "Unsolicited Notifications" che non sono risposte a richieste del client. La RFC 4511 definisce un unico tipo di "Unsolicited Notifications" che il server deve inviare a tutte le connessioni aperte quando sta per chiudersi.

Un metodo alternativo di rendere sicura la connessione è quello di usare un tunnel SSL. Per consuetudine questo è indicato con lo "scheme" *ldaps://* nella URL ma questa notazione non è standardizzata in nessuna RFC, anzi questo comportamento è deprecato fin dal 2003 nella RFC 3494 che ha ufficialmente abbandonato LDAPv2.

4.3.1 OpenLDAP

OpenLDAP è una implementazione open source del protocollo Lightweight Directory Access Protocol (LDAP), standard per l'interrogazione e la modifica di servizi di directory come un elenco aziendale di email o una rubrica telefonica o più in generale qualsiasi raggruppamento di informazioni che può essere espresso come record di dati ed organizzato in modo gerarchico.

LDAP è specificato in una serie di Standard Track Request for Comments (RFCs) della Internet Engineering Task Force (IETF). La sua descrizione in ASN.1 è attualmente pubblicata nella RFC 4511.

4.3.2 Directory LDAP

l termine di uso comune "*directory LDAP*" può essere fuorviante, in quanto LDAP definisce un protocollo d'accesso e non una base di dati. Nessun tipo specifico di directory è una "*directory LDAP*".

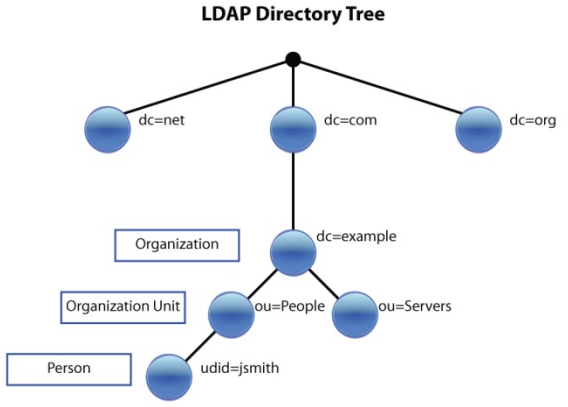
Si potrebbe ragionevolmente usare il termine per descrivere qualsiasi directory accessibile tramite LDAP e che possa identificare gli oggetti contenuti tramite nomi X.500, ma Directory come OpenLDAP e i suoi predecessori sviluppati presso l'Università del Michigan, anche se progettati espressamente per l'accesso tramite LDAP piuttosto che come ponte verso X.500, come avveniva per i prodotti forniti da

ISODE, non sono *directory* LDAP più di qualsiasi altra *directory* accessibile tramite protocollo LDAP.

4.3.3 Struttura

L'informazione all'interno di una directory è organizzata in elementi chiamati *entry*.

Gli elementi di una *directory LDAP* presentano una struttura gerarchica che riflette confini politici, geografici o organizzativi. Nel modello X.500 originale, gli elementi che rappresentano gli stati appaiono in cima all'albero, con sotto di essi gli elementi per gli stati federali o le organizzazioni nazionali (normalmente nelle installazioni di LDAP vengono usati i nomi del DNS per strutturare i livelli più alti della gerarchia). Più in basso potrebbero apparire elementi per rappresentare le divisioni all'interno di una singola organizzazione, singole persone, documenti, stampanti o qualsiasi altra cosa.



Nella struttura ad albero, ad ogni livello esiste un *Relative Distinguished Name* (RDN) che lo identifica (ad esempio *ou=people*). L'unione di tutti i RDN, presi in successione dal nodo foglia fino alla radice, costituisce il *Distinguished Name* (DN), una stringa che rappresenta univocamente una entry nella directory.

Un *dn* può essere ad esempio:

```
"cn=Antonio Messina,ou=Users,dc=pa,dc=icar,dc=cnr,dc=it"
```

Ciascuna entry ha una serie di attributi, costituiti dall'associazione attributo-valore; per ogni attributo possono esserci più valori. Ognuno degli attributi dell'elemento è definito come membro di una classe di oggetti, raggruppati in uno schema.

Ogni elemento nella directory è associato a una o più classi di oggetti, che definiscono se un attributo sia opzionale o meno, e che tipo di informazioni questo contenga.

I nomi degli attributi solitamente sono scelti per essere facilmente memorizzabili, per esempio "*cn*" per *common name*, o "*mail*" per un indirizzo e-mail.

I valori degli attributi dipendono dal tipo, e la maggioranza dei valori non binari sono memorizzati in LDAPv3 (LDAP versione 3) come stringhe UTF-8. Per esempio, un attributo di tipo mail potrebbe contenere il valore "user@example.com", mentre un attributo "jpegPhoto" potrebbe contenere una fotografia nel formato (binario) JPEG.

Nella definizione della classe dell'oggetto LDAP alcuni attributi sono obbligatori (MUST) mentre altri sono opzionali (MAY). Per ottenere la creazione dell'oggetto è indispensabile definire contemporaneamente tutti gli attributi obbligatori.

Ogni attributo ha una propria definizione che include anche una specifica del tipo di dato e delle regole di matching. È possibile definire attributi e classi di oggetti personalizzati.

È necessario effettuare un Check dopo ogni configurazione.

4.3.4 URL LDAP

Esiste un formato per una URL che identifica una operazione LDAP e che solitamente viene utilizzato per effettuare ricerche o per consentire al server di ritornare dei "*referrals*" cioè riferimenti ad un altro server che contiene le informazioni richieste dal client:

```
ldap://host:port/DN?attributes?scope?filter?extensions
```

Non tutte le parti della URL sono obbligatorie.

- *ldap://* indica lo scheme della URL ed identifica il protocollo LDAP
- *host* è il FQDN o l'indirizzo IP del server LDAP.
- *port* è la porta sulla quale contattare l'host
- *DN* è il nome completo ("distinguished name") utilizzato per identificare la base (il punto di partenza) della ricerca.
- *attributes* è la lista (con gli elementi separati da virgole) di attributi da recuperare.
- *scope* specifico l'ambito d'azione della ricerca (può essere "base", "one" oppure "sub").
- *filter* è il filtro di ricerca (come definito nella RFC 4515).
- *extensions* sono estensioni alla richiesta LDAP

È comunemente utilizzato uno scheme di tipo "*ldaps://*" che, sebbene deprecato nella RFC 3494, indica una connessione LDAP con SSL. Questo è completamente differente dall'utilizzo dell'operazione StartTLS che invece utilizza lo scheme standard *ldap://*.

4.3.5 Installazione e Configurazione su CentOS Linux 6.5

L'installazione del servizio OpenLDAP su di un server CentOS Linux 6.5 può essere effettuata rapidamente mediante l'utilizzo a console del tool *yum*:

```
# yum install openldap-servers openldap-clients
```

Per quanto riguarda l'attività di configurazione, si può seguire la serie di step presentati nei sotto-paragrafi seguenti.

4.3.6 ldap.conf

Il file */etc/openldap/ldap.conf* viene utilizzato da tutti i tools e librerie client-side:

```
BASE    dc=sicilia,dc=infse,dc=it
URI     ldap://127.0.0.1

TLS_CACERTDIR  /etc/openldap/certs
TLS_REQCERT    never
```

4.3.7 Copia dei file dalla directory */usr/share/openldap-server*

```
# cp /usr/share/openldap-servers/slapd.conf.obsolete /etc/openldap/slapd.conf
# cp /usr/share/openldap-servers/DB_CONFIG.example /var/lib/openldap/DB_CONFIG
```

4.3.8 Generazione della password di amministrazione

```
# slappasswd
New password: xxxxxxx
Re-enter new password: xxxxxxx
{SHA}xxxxxxxxxxxxxxxxxxxxxxxxxx
```

La password criptata così generata andrà inserita nel file di configurazione */etc/openldap/slapd.conf* in corrispondenza della direttiva *rootpw*.

4.3.9 Generazione certificati per SSL/TLS

Per abilitare l'interrogazione della directory ldap tramite protocolli LDAPS e TLS, è necessario generare il certificato che il demone slapd dovrà utilizzare:

```
# cd /etc/pki/tls/certs
# rm slapd.pem
# make slapd.pem
# chmod 640 slapd.pem
```

```
# chown :ldap slapd.pem
# ln –s /etc/pki/tls/certs/slapd.pem /etc/openldap/cacerts/slapd.pem
```

Il servizio slapd potrà partire con LDAPS attivato solo se:

- si aggiunge la riga

  SLAPD_LDAPS = yes

  nel file */etc/sysconfig/ldap*

- si inseriscono le direttive *TLSCACertificateFile*, *TLSCertificateFile* e *TLSCertificateKeyFile* nel file di configurazione */etc/openldap/slapd.conf*.

I tools e le librerie client-side di openldap si aspettano di trovare i dati relativi ai certificati nella directory indicata dalla direttiva *TLS_CACERTDIR* nel file */etc/openldap/ldap.conf*.

4.3.10 infse.schema

Poiché, per gli scopi del progetto, ad ogni persona da censire vanno associate anche le informazioni relative ai consensi che questa da per l'alimentazione e la consultazione del proprio fascicolo sanitario, è stata predisposta una object class ad-hoc denominata *utenteINFSE*, che estende l'object class standard *inetOrgPerson* mediante la presenza di due nuovi attributi:

- *consensoAlimentazione*: di tipo nativo booleano e a valore singolo;
- *consensoConsultazione*: di tipo *distinguishedName* e multi valore.

Queste definizioni sono state inserite nel file */etc/openldap/schema/infse.schema*.

```
# ------------------------------------------------
#       macros
# ------------------------------------------------
#
objectIdentifier icarROOT            1.3.6.1.4.1.39840
objectIdentifier icarLDAP            icarROOT:2
objectIdentifier infse               icarLDAP:1
objectIdentifier infseAttributeType  infse:1
objectIdentifier infseObjectClass    infse:2

# ------------------------------------------------
#       attribute definitions
# ------------------------------------------------
#
attributetype ( infseAttributeType:1
        NAME 'consensoAlimentazione'
        EQUALITY booleanMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
        SINGLE-VALUE )

attributetype ( infseAttributeType:2
        NAME 'consensoConsultazione'
        SUP distinguishedName )

# ------------------------------------------------
#       object class definitions
# ------------------------------------------------
#
objectclass ( infseObjectClass:1
```

```
        NAME 'utenteINFSE'
        DESC 'Utente generico INFSE'
        SUP inetOrgPerson
        MAY ( consensoAlimentazione $ consensoConsultazione ) )
```

4.3.11 slapd.conf

Nella predisposizione del file di configurazione */etc/openldap/slapd.conf* si è posta particolare attenzione all'attivazione della funzionalità di *Reverse Group Membership*, che consente di determinare a quali gruppi appartiene una entry senza eseguire un'ulteriore altra ricerca.

Si è quindi esplicitato il caricamento del modulo *memberof.la* ed attivato il meccanismo di overlay per l'attributo dinamico *memberOf*.

Da notare che, oltre agli schema standard di openldap, viene anche richiesto l'utilizzo del nuovo schema *infse.schema*.

```
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.

include         /etc/openldap/schema/corba.schema
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/duaconf.schema
include         /etc/openldap/schema/dyngroup.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/java.schema
include         /etc/openldap/schema/misc.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/openldap.schema
include         /etc/openldap/schema/ppolicy.schema
include         /etc/openldap/schema/collective.schema

include         /etc/openldap/schema/infse.schema


# Allow LDAPv2 client connections.  This is NOT the default.
allow bind_v2

pidfile         /var/run/openldap/slapd.pid
argsfile        /var/run/openldap/slapd.args

# Load dynamic backend modules
modulepath /usr/lib64/openldap
moduleload memberof.la

# The next three lines allow use of TLS for encrypting connections using a
# dummy test certificate which you can generate by running
# /usr/libexec/openldap/generate-server-cert.sh. Your client software may balk
# at self-signed certificates, however.

TLSCACertificateFile /etc/pki/tls/certs/ca-bundle.crt
TLSCertificateFile /etc/pki/tls/certs/slapd.pem
TLSCertificateKeyFile /etc/pki/tls/certs/slapd.pem


# enable on-the-fly configuration (cn=config)
database config
rootdn          "cn=admin,cn=config"
rootpw           {SSHA}yfMQKONR3rrA17Nspx7cOvOWcQ4d/3L+
access to *
        by dn.exact="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" manage
        by * none

# enable server status monitoring (cn=monitor)
```

```
database monitor
access to *
       by dn.exact="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read
        by dn.exact="cn=Manager,dc=sicilia,dc=infse,dc=it" read
        by * none

#######################################################################
# database definitions
#######################################################################

database       bdb
suffix         "dc=sicilia,dc=infse,dc=it"
checkpoint     1024 15
rootdn         "cn=Manager,dc=sicilia,dc=infse,dc=it"
rootpw         {SSHA}yfMQKONR3rrA17Nspx7cOvOWcQ4d/3L+

access to dn.subtree="dc=sicilia,dc=infse,dc=it"
    by self write
    by  set="[cn=Administrators,ou=Groups,dc=sicilia,dc=infse,dc=it]/member*  &  user"
write
    by set="[cn=Operators,ou=Groups,dc=sicilia,dc=infse,dc=it]/member* & user" read
    by * break

access to attrs=userPassword
  by anonymous auth
  by self =rwdx
  by  set="user  &  [cn=Administrators,ou=Groups,dc=sicilia,dc=infse,dc=it]/member*"
manage
  by dn.children="ou=Special Accounts,dc=sicilia,dc=infse,dc=it" auth

# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd and slap tools.
# Mode 700 recommended.
directory      /var/lib/ldap

# Indices to maintain for this database
index objectClass                       eq,pres
index ou,cn,mail,surname,givenname      eq,pres,sub
index uidNumber,gidNumber,loginShell    eq,pres
index uid,memberUid                     eq,pres,sub
index nisMapName,nisMapEntry            eq,pres,sub

overlay memberof

cachesize 10000
```

4.3.12 root.ldif

Prima di eseguire per la prima volta il demone slapd, bisogna creare l'oggetto root della directory. A tale scopo va preparato un file di testo, denominato in questa sede *root.ldif*, contenente la definizione di oggetti in formato *LDAP Data Interchange Format* (LDIF):

```
dn: dc=sicilia,dc=infse,dc=it
objectClass: dcObject
objectClass: organization
dc: sicilia
o: sicilia

dn: ou=Groups,dc=sicilia,dc=infse,dc=it
ou: Groups
objectClass: organizationalUnit

dn: ou=Users,dc=sicilia,dc=infse,dc=it
ou: Users
objectClass: organizationalUnit
```

Oltre all'oggetto root che costituirà il distinguished name di base, sono qui definite anche due unità organizzative che andranno a contenere utenti e gruppi.

4.3.13 Inizializzazione ed avvio del servizio di directory

Per inizializzare la directory ldap, si deve rimuovere l'eventuale contenuto della directory /etc/openldap/slapd.d e quindi caricare il contenuto del file root.ldif predisposto precedentemente:

```
# rm –rf /etc/openldap/slapd.d/*
# slapadd –n 2 –l /root/root.ldif
# slaptest –f /etc/openldap/slapd.conf –F /etc/openldap/slapd.d
```

A questo punto, assegnate le ownership delle directory, si può attivare e far partire il servizio:

```
# chown –R ldap:ldap /var/lib/ldap
# chown –R ldap:ldap /etc/openldap/slapd.d
# chkconfig –level 235 slapd on
# service slapd start
```

4.4 Shibboleth

Shibboleth è un pacchetto software opensource e standardizzato che ha per obiettivo il Web Single Sign-On e si basa, prevalentemente, sullo standard *Security Assertion Markup Language* (SAML).

SAML è uno standard informatico per lo scambio di dati di autenticazione ed autorizzazione (dette asserzioni, in formato XML) tra domini di sicurezza distinti, tipicamente un *Identity Provider* (IdP), entità che fornisce informazioni di identità, e un *Service Provider* (SP), entità che fornisce servizi.

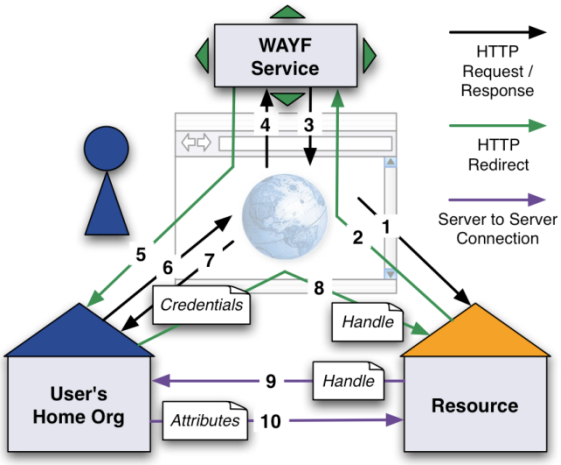
Le asserzioni SAML possono contenere tre tipi di informazioni:

- *Authentication statement*: indica che l'utente è stato autenticato;
- *Attribute statement*: indica che un utente è associato con gli attributi specificati. Un attributo è una coppia (nome, valore);
- *Authorization decision statement*: specifica che un utente ha l'autorizzazione a compiere un'azione su una risorsa.

Il processo di autenticazione avviene sostanzialmente nei seguenti termini:

1. l'utente richiede la connessione ad una risorsa protetta facendo sì che il SP intercetti la richiesta (la risorsa da proteggere è definita nei files di configurazione del web server che la ospita);

2. il SP, basandosi sulla configurazione di cui sopra, determina a quale IdP far riferimento e quale protocollo utilizzare attraverso un meccanismo di "scoperta" noto come servizio *WAYF* (acronimo di Where Are You From): la richiesta di autenticazione è così delegata al WAYF che la passa all'IdP selezionato dall'utente;



3. come risultato delle azioni precedenti, una richiesta di autenticazione via browser è inviata dal SP all'IdP selezionato dall'utente: l'IdP decide se l'utente può autenticarsi e quali attributi inviare al SP, scelta questa basata prevalentemente sulle caratteristiche del SP che fornisce il servizio e su quelle dell'attributo principal dell'utente;

4. l'IdP impacchetta e firma i dati che trasmette sotto forma di asserzione SAML al SP che la spacchetta e la decodifica eseguendo poi una serie di controlli di sicurezza per decidere infine se il richiedente abbia o meno diritto di accesso alla risorsa desiderata;

5. infine, se il processo di verifica del punto precedente ha esito positivo, l'utente viene finalmente rediretto alla risorsa richiesta.

Come si può evincere da quanto sopra esposto, il processo di autenticazione avviene sull'IdP del richiedente e non sul SP che tale risorsa distribuisce; in altre parole, l'utente si autentica presso la sua propria organizzazione, utilizzando una sola coppia di credenziali per più servizi ed evitando di dover allocare una coppia di credenziali per ogni fornitore di servizi.

Affinché ciò avvenga, occorre che gli attori del processo siano consorziati in quelle che vengono dette *Federazioni*, istituzioni collaborative di fornitori ed utilizzatori di servizi.

Una implementazione JAXR é detta JAXR Provider e può essere di livello 0 o livello 1. Questo indica quali elementi sono supportati dall'implementazione. Ad esempio, il supporto ad UDDI é di livello 0. Questo significa che qualsiasi prodotto che dichiari la compatibilità a JAXR dovrà per forza supportare l'accesso ad UDDI. Il supporto ebXML é invece posto a livello 1, quindi i prodotti JAXR level 1 complaint supporteranno sia UDDI che ebXML.

4.5 Central Authentication Service (CAS)

Il *Central Authentication Service* (CAS) permette di realizzare un sistema di autenticazione unica, *Single Sign On* (SSO), tra applicazioni basate su interfaccia web o mail server, per mezzo di ticket di autenticazione. Le applicazioni demandano il problema dell'autenticazione al server centrale CAS, senza più preoccuparsi dei vari problemi di identificazione dell'utente.

CAS fu creato inizialmente come progetto open source dal gruppo Information Tecnology Services della Yale Universities; successivamente divenne un progetto JA-SIG, sempre in cooperazione con Yale e con la Reutgers Universities. È utilizzato da molte università americane ed europee integrando i sistemi di autenticazione preesistenti basati su LDAP o Kerberos.

Il sistema CAS è multipiattaforma, dato che il server è implementato su servlet java, e può essere eseguito su di un servlet engine, seguendo le specifiche JSP versione 1.2. Le librerie per il server applicativo sono inoltre disponibili per vari linguaggi di sviluppo o anche di scripting, come JAVA, JSP, Perl, PHP, ASP, PL/SQL, Python.

l sistema CAS nella sua forma più generale è composto principalmente da tre diverse componenti:

- il server CAS centrale
- un server applicativo che fornisce un servizio
- un client utente, che intende accedere al servizio applicativo, solitamente tramite un browser web.

Alle volte a queste tre componenti se ne può aggiungere una quarta, negli specifici casi in cui un ulteriore applicazione deve autenticarsi impersonificando l'utente vero e proprio. Questa modalità di funzionamento è detta modalità proxy.

Il server CAS centrale è il componente preposto all'identificazione dell'utente, nonché il software che deve gestire una corretta e soprattutto sicura autenticazione. Il software e tutte le sue librerie server vengono installate e configurate su di un web server come Apache Tomcat, che utilizzi i protocolli HTTP e HTTPS.

Il server applicativo invece è una qualsiasi risorsa o servizio web-based, o una qualsiasi interfaccia web, che utilizzi anch'essa i protocolli HTTP o HTTPS, sul quale devono essere installate specifiche librerie CAS, apposite per le applicazioni.

Queste applicazioni che utilizzano il meccanismo di autenticazione proprio del CAS, vengono chiamate "*CAS-ifyed*".

Il client utente o semplicemente l'utente, è il soggetto che intende accedere ai servizi del server applicativo, solitamente tramite un browser web. Per una configurazione ottimale, il browser dell'utente dovrebbe aver abilitato i cookie e i redirect Javascript, anche se questa configurazione non è strettamente necessaria per un corretto funzionamento del sistema CAS.

4.5.1 Caratteristiche di CAS

Il CAS fornisce un sistema in SSO basato su di un protocollo aperto e ben documentato. È dotato di un componente server Java open source e dispone di una buona documentazione e soprattutto di una vasta comunità di sviluppatori.

Gli scopi principali del server CAS sono:

- facilitare l'autenticazione centralizzata SSO per più applicazioni comuni;
- semplificare la procedura stessa di autenticazione;
- consentire un autenticazione sicura agli utenti, senza che questi debbano rivelare le proprie credenziali ai servizi e agli applicativi ogni volta.

Le caratteristiche migliori del sistema CAS sono sicurezza, affidabilità, flessibilità. La sicurezza dell'intero sistema risulta notevolmente rafforzata, utilizzando i meccanismi di CAS. Infatti per l'intero ciclo di autenticazione, le credenziali dell'utente viaggiano solo una volta tra il browser dell'utente ed il server, e su di un canale protetto e crittografato SSL.

Durante l'autenticazione, il server CAS cerca di salvare nel browser dell'utente uno speciale cookie privato: il *Ticket Granting Cookie* (TGS). Questo non contiene nessun tipo di dato confidenziale, per questo viene denominato opaco, ma contiene solamente un identificativo di sessione noto al CAS. Transita sulla rete in chiaro senza protezione, ma solo il CAS può leggerlo o scriverlo. Il ticket ha comunque un tempo di vita relativamente breve, e viene cancellato definitivamente alla chiusura del browser da parte dell' utente.

Questo speciale ticket, serve come garanzia di già avvenuta autenticazione, qualora l'utente intenda accedere ad un'altra applicazione del sistema. In questo modo l'utente in tutta trasparenza non deve re autenticarsi ogni volta che cambia il servizio applicativo richiesto.

Le applicazioni invece non comunicano attraverso il ticket TGT, ma attraverso altri ticket appositi, detti *Service Ticket* (ST). Anch'essi sono opachi, non contenendo informazioni personali, e possono essere utilizzati solo una volta (one time ticket).

Il ST viene emesso dal server CAS verso l'utente, il quale in modo automatico lo passa al servizio applicativo. Questi lo ridirige di nuovo verso il CAS per confermare l'identità dell'utente, chiudendo il ciclo di autenticazione.

Per quanto riguarda la flessibilità, nel pacchetto software che costituisce il server CAS sono comprese tutte le funzionalità per la una corretta gestione del suo protocollo.

Invece la procedura che verifica l'identità dell'utente è lasciata completamente allo sviluppatore, garantendogli una notevole libertà di scelta su come adattare il server ai propri meccanismi di autenticazione locale, tramite plugin per un Database relazionale, un server LDAP, certificati X509, semplici file di testo protetti da password, o anche una combinazione di questi impostando una scala di priorità da seguire.

Inoltre i plugin possono essere estesi per gestire meccanismi già esistenti come Kerberos o Active Directory. Per questo sono disponibili varie librerie di sistema, con le quali lo sviluppatore può interfacciare il server CAS con il metodo di autenticazione da lui preferito.

Un'altra peculiarità del CAS è il suo possibile uso in modalità *Proxy*. Un proxy è un servizio, che a sua volta deve impersonare un utente autenticandosi su altri Servizi.

Questa modalità è stata introdotta con la seconda versione di CAS, perché in generale in un sistema SSO non si dovrebbe limitare solamente ad un'autenticazione diretta tra un client ed un server, ma a volte coinvolgendo una terza parte. Per esempio un servizio interpellato da un utente potrebbe dover contattare un altro servizio back-end, o utilizzando un portale, l'utente autenticato avrebbe la necessità di interrogare un'applicazione esterna.

Questi sistemi terzi quindi devono a loro volta poter identificare l'utente, e non sarebbe ovviamente nè saggio nè comodo reinviare a questi le credenziali personali dell'utente per identificarlo. Per questo il server CAS permette l'utilizzo di speciali ticket per questa modalità, chiamati *Proxy Garantig Ticket* (PGT). Questi permettono un'autenticazione sicura su applicazioni terze, garantendo a queste l'identità dell'utente.

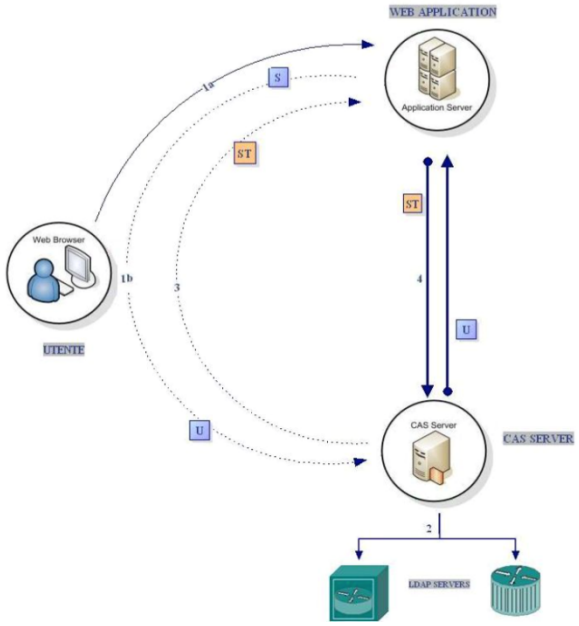
Rispetto ad un normale sistema SSO, il CAS ha inoltre delle caratteristiche specifiche.

Le comunicazioni tra client e server, e tra server centrale e applicativo, viaggiano tutte sul protocollo protetto HTTPS, e vengono utilizzati solamente i certificati digitali generati e gestiti dalla CA specificata all'interno della configurazione del server CAS centrale.

Questi due aspetti contribuiscono a una doppia sicurezza: le comunicazioni sono crittografate e riservate, e l'identità dei soggetti è garantita. Gestendo i certificati digitali in modo centralizzato, è quindi possibile creare un'infrastruttura a chiave pubblica interna e locale.

4.5.2 CAS in modalità diretta

Questo schema di funzionamento non prevede terze parti nel sistema, e viene attivata nel momento in cui un utente tramite browser, cerca di accedere ad un servizio applicativo la cui autenticazione è gestita dal CAS. Questa è la richiesta iniziale [1a]. Il server applicativo rimanda l'utente alla pagina di login del server CAS, [1b], tramite connessione sicura HTTPS, passando come parametro aggiuntivo l'URL dell'applicazione richiesta.



Il CAS chiede all' utente che gli vengano fornite le sue credenziali, tramite la relativa form di autenticazione o tramite l'invio automatico del certificato digitale utente, se disponibile. Queste vengono controllate e verificate nel passo [2], con il servizio di autenticazione scelto, LDAP in questo caso. Tutte le comunicazioni tra utente e server CAS passano attraverso un canale SSL sicuro e crittografato. Se l'autenticazione non ha avuto successo verrà visualizzato un opportuno messaggio contenente l'errore ottenuto, come nome utente o password incorretti, e l'utente non avrà accesso al servizio. In caso invece di autenticazione avvenuta, l'utente viene ridiretto automaticamente all'applicazione richiesta [3], appendendo però all'URL di redirezione un ticket, chiamato service ticket (ST), sotto forma di una lunga stringa alfanumerica. Questo ticket serve per indicare che l'utente ha eseguito correttamente l'autenticazione, e richiesto quella specifica applicazione. Infatti il server CAS nel momento in cui verifica le credenziali dell'utente, prende anche nota del servizio da

lui richiesto, così da associare al ticket stesso, il nome dell'utente e il servizio richiesto. Il service ticket è utilizzabile una sola volta, ed invalidato non appena utilizzato.

Inoltre il CAS cerca di salvare nel browser dell'utente un cookie, il Ticket Granting Cookie (TGC) valido per la sessione globale, ed eliminato automaticamente dopo un intervallo di tempo o dopo la chiusura del browser stesso. Questo meccanismo permette di identificare l'utente senza doversi riautenticare, nel caso in cui questi richiedesse di accedere ad un'altra applicazione del sistema, implementando quindi la tecnica del SSO.

A questo punto l'applicazione richiamata dall'utente, verifica se il service ticket passatogli dal browser utente sia corretto, tramite una comunicazione protetta HTTPS con il server CAS (passo [4]). Il server CAS controlla che il ticket inviatogli dall'applicazione sia valido e associato allo stesso servizio richiesto, dopodiché se la validazione ha successo, viene restituito al chiamante il nome utente. A questo punto l'applicazione è sicura dell'identità dichiarata dell'utente, e può procedere nella propria comunicazione con esso, chiudendo di fatto il ciclo di autenticazione.

L'annullamento della sessione globale può essere eseguita semplicemente accedendo alla pagina di logout del server CAS. Questa non farà altro che eliminare il cookie TGC. Altrimenti questo viene annullato automaticamente alla chiusura del briose da parte dell'utente, oppure dopo un intervallo temporale di circa un'ora.

Utilizzando una metodologia di autenticazione come questa, l'utente ha un grado di sicurezza molto elevato, infatti le proprie credenziali hanno viaggiato sulla rete solo una volta in tutto il ciclo di autenticazione, peraltro su un canale sicuro. Inoltre le applicazioni sono completamente trasparenti, infatti l'identificazione ad esse è basata su ticket opachi non contenenti ne le credenziali utente ne informazioni riservate.

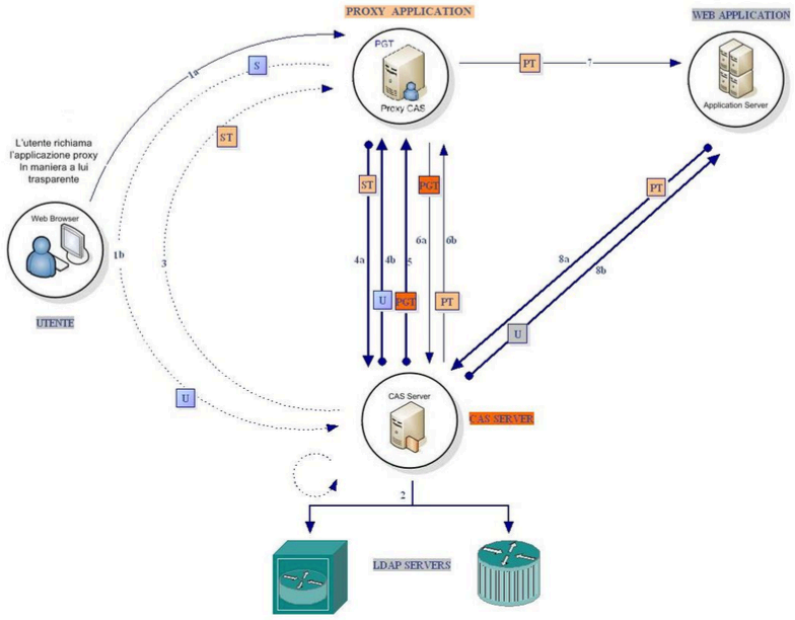
Oltre a ciò anche le re-autenticazioni seguenti vengono affrontate in maniera efficace, se l'utente aveva precedentemente accettato il cookie TGC: al momento di una richiesta di un nuovo servizio applicativo, non vedendo un ticket ST valido, rimanderà l'utente al server CAS. Questi leggendo il cookie TGC riconoscerà automaticamente l'utente e la sua sessione attiva, quindi creerà per esso un nuovo ticket ST specifico per il nuovo servizio, il tutto in modo completamente trasparente all'utente.

4.5.3 CAS in modalità proxy

Gli sviluppatori di CAS a Yale hanno pensato di introdurre nella versione 2.0 alcune nuove caratteristiche tra le quali la possibilità di sviluppare una applicativo proxy e su più livelli detta anche multitier, restando però compatibile con la versione precedente. Questa modalità, detta anche di tipo CAS-proxy, è necessaria quando un'applicazione

deve fare le veci di qualcun'altro (un utente o un'altra applicazione) nell'autenticazione, cioè essa stessa deve autenticarsi su di un'altra applicazione impersonificando un utente. Un tipico esempio nel quale è necessario un meccanismo di questo tipo è un'applicazione webmail in un sistema in SSO.

L'utente si autenticherà tramite CAS, e accederà poi a webmail essendosi già autenticato. Questa applicazione però dovrà collegarsi al server IMAP della posta per poterla leggere, facendo quindi le veci dell'utente.



Per un corretto funzionamenti di questa modalità è necessario introdurre tre ulteriori tipologie di ticket, oltre ai già noti ticket TGC e ST:

- *Proxy-granting ticket* (PGT): è un ticket mandato dal Server CAS ad una applicazione in possesso di un ticket ST valido, associato al singolo utente e a quella determinata applicazione. Conferisce a questa il permesso di richiedere dei proxy-ticket (PT), diventando di fatto un Proxy.
- *Proxy-granting ticket IOU* (PGTIOU): è un ticket mandato dal Server CAS durante il processo di validazione tra CAS e Proxy, insieme a un PGT. Il proxy ha il compito di gestire una tabella che correli i ticket PGTIOU e PGT.
- *Proxy ticket* (PT): è un ticket utilizzabile da un proxy per accedere ad una applicazione esterna impersonando l'utente. Un ticket PT identifica il proxy o la catena di Proxy che hanno richiesto l'autenticazione al servizio esterno, infatti, dato che questo è un meccanismo riproducibile a catena, un PT può essere utilizzato da altri proxy per ottenere a sua volta un PGT. Viene inoltre conservata una traccia della catena di proxy, cioè del percorso di autenticazione, tra l'utente e applicazione esterna finale.

Il funzionamento di CAS in versione proxy, è analogo alla modalità diretto fino al passo [4]. Su richiesta dell'applicazione, il server CAS fornisce un elemento aggiuntivo nella comunicazione in formato XML [5], un ticket PGTIOU.

Questo è un valore identificativo con cui è indicizzato il corrispondente PGT, il Proxy Granting Ticket, che viene creato dal server CAS. Il PGT viene quindi trasmesso [5] al server proxy, insieme al corrispondente PGTIOU. Questa comunicazione è protetta tramite un canale SSL, utilizzando i certificati digitali dei due server, che devono essere opportunamente installati, per garantire la segretezza del PGT. Questo tipo di comunicazione è necessaria anche per garantire a ciascuna delle due controparti, CAS server e proxy server, l'identità dell'altra, realizzando una mutua autenticazione, in un modo semplice e funzionale. Il PGT viene memorizzato autonomamente, e mappato al PGTIOU dal servizio proxy per usarlo successivamente. Il PGT permette infatti al proxy di richiedere, anche più volte, dei Proxy Ticket PT, tramite i quali può impersonificare l'utente alla specifica applicazione web richiesta. Infatti un ticket PGT è legato ad un utente, ad un proxy e a più servizi applicativi, mentre un ticket PT è legato ad un utente, ad un proxy e ad un servizio applicativo specifico, e può essere utilizzato una sola volta.

A questo punto quindi il proxy ha bisogno del ticket PT associato all'applicazione web nella quale deve impersonificare l'utente: questo viene richiesto al server CAS nel passo [6a] attraverso una comunicazione HTTPS, passando come parametro il ticket PGT ed il servizio di destinazione. Il server CAS vede questa richiesta, verifica la validità del PGT, genera il PT richiesto, e lo spedisce al proxy ([6b]).

Il server proxy può ora quindi comunicare questo PT all'applicazione che intende utilizzare [7], nelle stesse modalità in cui il browser dell'utente si collega con un servizio applicativo nel funzionamento diretto.

A sua volta il servizio verifica il PT comunicando in HTTPS con il server CAS [8a], mandando come argomenti il suo ID di servizio e il PT, ottenendo come risposta [8b] l'Username dell'utente autenticato e la traccia della catena dei proxy coinvolti, se ovviamente il PT è valido. Il Target può inoltre decidere, alla luce di questi dati, come e se dare accesso al Proxy, e procedere come più opportuno.