IdeaManager, un'applicazione “distribuita” per l’Identity Management

R. Conte, F. Sansone, A.P. Pala, A.M. Teodorescu, R. Guarino

Istituto di Fisiologia Clinica, Consiglio Nazionale delle Ricerche

Introduzione

La gestione delle identità digitali è un processo che interessa ogni organizzazione dotata di un sistema informativo. Pur non avendo procedure formali e/o sistemi dedicati, le identità digitali sono comunque gestite. Applicare delle best-practices nella gestione delle identità digitali riduce i costi per l’organizzazione e migliora l’efficenza della stessa, consentendo inoltre di limitare i rischi oltre che rispettare le normative legate alla privacy dei dati personali e/o sensibili.

È noto che un’identità digitale possa essere rappresentata come un insieme di attributi e che tale insieme possa essere suddiviso in sottoinsiemi, ognuno dei quali rappresenta un’identità parziale dell’individuo [Borcea-Pfitzman et al. 2006].

Nel tempo la gestione delle identità digitali si è evoluta ed è possibile infatti identificare in tale processo alcune fasi principali: una sorta di età della pietra vede le applicazioni gestire il nucleo minimo ed essenziale per l’autenticazione in maniera totalmente autonoma e indipendente; successivamente si sviluppano i primi sistemi per centralizzare le credenziali ed un nucleo di attributi sugli utenti (es. Kerberos, LDAP), anche se le applicazioni gestiscono autonomamente le informazioni necessarie per il processo di autorizzazione; infine le credenziali e l’insieme delle informazioni che caratterizzano l’identità digitale di un utente sono centralizzate (negli Identity Management System) e le applicazioni possono gestire solo gli “app-specific user data”. L'Identity Manager, oltre che centralizzare il processo di autenticazione, diventa di supporto anche al processo di autorizzazione fornendo al SP gli attributi sui quali questo può decidere se, ed a quale livello, autorizzare l’accesso ai dati (Attribute/Role Based Access Control).

Le architetture federate per l’autenticazione e l’autorizzazione portano questo concetto oltre i limiti dell’organizzazione e tramite le virtual organisation è possibile accedere a servizi sulla base di attributi aggregati in tempo reale forniti dalla propria e da organizzazioni esterne.

Tali tecnologie introducono quindi dei significativi vantaggi per l’utente, che può disporre di numerosi servizi offerti dalla propria o da altre organizzazioni, utilizzando un unico meccanismo di autenticazione ed un’unica identità aggiornata allo stato corrente riguardo il rapporto con tali organizzazioni. Tale semplificazione favorisce a sua volta la creazione di sistemi informativi autonomi, sviluppati per specifici obiettivi o flussi di informazioni (es. presenze, gestione progetti, produzione scientifica, risorse assegnate ecc.), con l’effetto collaterale che tutte le informazioni che riguardano l’individuo (le identità parziali) vengono distribuite su sistemi diversi, con interfacce utente diverse, generando confusione nell’utente che a volte non è neppure consapevole dell’esistenza di particolari strumenti.

IdeaManager si propone quindi come collettore di quell’insieme significativo (per l’utente) di informazioni che riguardano un individuo all’interno di un’organizzazione anche quando queste provengano da sorgenti esterne all’organizzazione stessa.

Il contesto

Il progetto si colloca nel contesto della gestione delle informazioni relative al curriculum del dipendente dell’Istituto di Fisiologia Clinica del CNR, cercando di dare una soluzione alla necessità del personale di poter disporre delle informazioni inerenti la propria attività lavorativa in maniera completa ed integrata. Allo stesso tempo il progetto ha come obiettivo la gestione

del ciclo di vita delle identità digitali associate al personale, utilizzate per l'intero percorso lavorativo (sia esso un dipendente, uno studente o un semplice frequentatore) per lo svolgimento della propria attività o delle procedure amministrative, tecniche o di governo. Spesso avviene infatti che le informazioni siano disponibili ma difficilmente accessibili all'utente finale perché prodotte e gestite con differenti software, con differenti credenziali e diritti di accesso e in alcuni casi con restrizioni non necessarie a causa di una gestione non integrata degli stessi diritti.

Nonostante sia stata progettata in un contesto ben definito l'applicazione è stata pensata e sviluppata in maniera da essere agnostica rispetto ad esso con l'obiettivo di essere distribuita come software open-source.

Strumenti e metodi

L'applicazione si basa su un'architettura modulare (realizzata tramite la metodologia REST[1]), in cui una WebApp, con cui si realizza il frontend per l'utente, si interfaccia ad un backend *distribuito*.

REST definisce un modello per la progettazione di architetture software in cui il World Wide Web è visto come una piattaforma per l'elaborazione distribuita dei dati con tutto ciò di cui si ha bisogno, ossia l'infrastruttura, basata sul protocollo HTTP, e le informazioni, che diventano vere e proprie risorse. In particolare "REST emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems". Di conseguenza sviluppare nuove applicazioni RESTful o sviluppare delle interfacce REST per le applicazioni esistenti, consente una maggiore scalabilità delle applicazioni, un riutilizzo di quelle già sviluppate, una maggiore semplicità di manutenzione ed un maggiore coinvolgimento delle strutture periferiche di una organizzazione, nello sviluppo dell'intero sistema informativo. In tal modo quindi l'insieme degli attributi che vanno a costituire l'identità digitale non viene costruito duplicando e centralizzando i dati recuperati sulle diverse sorgenti remote, con conseguenti problemi di sincronizzazione, ma recuperandoli ed aggregandoli "on-demand". Questo di fatto sposta al livello dell'applicazione frontend appunto l'aggregazione dei dati, funzione che in passato veniva eseguita sui database relazionali tramite le operazioni di join. In questo modo ogni modulo del backend distribuito può gestire il dataset minimo di informazioni strettamente necessarie per lo scopo per cui è stato sviluppato, utilizzando il modello più appropriato per l'organizzazione dei dati (relazionale, document-oriented, graph, key-value), eventualmente tramite database NoSQL.

Poter disporre di un insieme di informazioni ottenute in tempo reale da diverse sorgenti consente inoltre di sviluppare una sorta di cruscotto di governo (dashboard) per l'accesso all'informazione completa ed in tempo reale, riguardante il singolo membro dell'organizzazione o l´intera comunità.

Il backend è quindi costituito da un modulo "core" (come wrapper di un server LDAP) e da altri moduli corrispondenti ai diversi sistemi informativi (presenze, gestione rendicontazione, pubblicazioni scientifiche ecc.) che espongono i propri dati come risorse attraverso API RESTful.

Il modulo core, grazie all'utilizzo di LDAP, consente di definire nuovi attributi semplicemente indicandoli in un file di configurazione YAML[2] (purché in LDAP sia già presente il corrispondente

---

[1] REpresentational State Transfer, metodologia teorizzata da Roy Fielding, fra i principali autori delle specifiche HTTP, nella propria tesi di PhD in Filosofia Informatica: [http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[2] http://www.yaml.org

schema). La definizione di un nuovo attributo si ottiene definendone anche le proprietà (*type*, *single* o *multivalue*, *optional* ecc.)[3] ed i diritti di accesso in funzione dei ruoli (alcuni predefiniti quali *self*, *all*, *none*). In funzione dell'utente che accede e del ruolo ad egli associato il core backend restituisce il set dei dati (ad oggi serializzato utilizzando il formato JSON) con le proprietà necessarie perché il frontend possa *renderizzarlo* ed eventualmente consentirne la modifica eseguendone la relativa verifica sintattica.

Gli attributi ricevuti dal core backend, con le relative proprietà, devono comunque essere "accettati" dal frontend. Ciò avviene anche in questo caso tramite file di configurazione, nei quali l'attributo viene definito associandogli le etichette per la localizzazione ed il formato di rappresentazione (*radiobutton*, *checkbox*, *menu* ecc.). Il frontend è sviluppato come WebApp (HTML5, CSS3, JavaScript) e fa uso di Bootstrap[4] per semplificare il mantenimento della GUI. La WebApp è pensata come un framework con la possibilità di gestire estensioni di primo livello (funzionalità applicabili sull'intero insieme di dati, come ad esempio statistiche, o di servizio, ad esempio notifiche) e di secondo livello. Tipicamente un'estensione ha l'obiettivo di visualizzare dati provenienti da un modulo backend diverso dal core (es. per la gestione delle presenze o la raccolta delle pubblicazioni scientifiche di un utente).

La scelta di utilizzare il formato YAML per i file di configurazione deriva dal fatto che la sintassi risulta facilmente comprensibile da un essere umano ma è allo stesso tempo supportata da diversi linguaggi di programmazione e ciò consentirà in futuro di poter sviluppare delle interfacce utente tramite cui poter compilare tali file.

Tutti i componenti dell'applicazione (frontend e moduli backend con API REST) sono di fatto delle WebApplication, di conseguenza possono sfruttare i meccanismi di autenticazione federata ed il SSO.

Conclusioni

L'obiettivo del progetto è quello di mappare la struttura dell'intero sistema informativo di un'organizzazione con la sua effettiva architettura, definita spesso dall'insieme di diversi e remoti sistemi informatici. L'aggregazione in tempo reale consente una maggiore consistenza e coerenza, oltre che l'immediata disponibilità dei dati non appena questi sono inseriti nel sistema informatico di uno specifico ufficio. Infine, gli stessi utenti possono facilmente contribuire all'inserimento o all'aggiornamento delle informazioni che li riguardano migliorando quindi la correttezza e completezza dell'intera base dati distribuita.

L'utilizzo di un'architettura REST, con moduli specializzati che "dialogano" tramite protocollo HTTP, consente la facile integrazione di sistemi e applicazioni esistenti, senza la necessità di dover modificare le interfacce grafiche o riprogettare le applicazioni. Ma anche lo sviluppo di applicazioni ex-novo risulta semplificato richiedendo il solo sviluppo dell'applicazione backend, con relativa interfaccia RESTful, oltre che di un modulo (che utilizza gli elementi grafici predefiniti) come estensione dell'applicazione frontend.

---

[3] Il core backend è pensato per l'utilizzo anche di database NoSQL, quindi schemaless, invece che LDAP.

[4] http://getbootstrap.com