



UNIVERSITA' DEGLI STUDI DI BARI

FACOLTA' DI SCIENZE MATEMATICHE, FISICHE E NATURALI

Corso di Laurea Specialistica in Informatica

TESI DI LAUREA

in

Sistemi per la Collaborazione in Rete

**Linked Data nello sviluppo collaborativo del software:
il caso di Hackystat**

Relatori:

Chiar.mo Prof. Filippo LANUBILE

Laureanda:

Myriam LEGGIERI

Indice

Introduzione	5
---------------------	----------

Capitolo I

1 Lo sviluppo del Software	6
-----------------------------------	----------

1.1 Pianificazione del lavoro	9
1.1.1 Project design	11
1.1.2 Build e unit test	12
1.1.3 Test integrati sulla QA	14
1.2 Rallentamenti nel ciclo di sviluppo	15
1.2.1 Issue	15
1.2.2 Scelte implementative	16
1.2.3 Utilizzo di tool e librerie	18
1.2.4 Scenario	20
1.2.4.1 Ricerca tramite codase	20
1.2.4.2 Ricerca tramite koders	24
1.2.4.3 ricerca tramite google code search	26
1.2.4.4 ricerca tramite google	27
1.2.5 Limiti dei motori di ricerca	
1.2.6 Limiti dei motori di ricerca semantici	
1.3 Soluzione proposta: profili collegati e descrizioni di dataset	32

Capitolo II

2 Evoluzione verso i Linked Data	34
---	-----------

2.1 Evoluzione del Web	35
2.1.1 Prima fase: il Web 1.0	35
2.1.2 Seconda fase: il Web 2.0	36
2.1.2.1 Filosofia open source	38

2.1.2.2	Intelligenza collettiva	39
2.1.2.3	Basi di dati proprietarie	39
2.1.2.4	Dagli Open ai Linked Data	40
2.1.3	Web Semantico	40
2.1.3.1	Tecnologie principali	42
2.1.3.2	Ostacoli posti alla diffusione	44
2.1.3.2.1	Rappresentazione della conoscenza	45
2.1.3.2.2	Confronto con il Web 2.0	48
2.1.4	Linked Data	50
2.1.4.1	Superamento dei data silos	51
2.1.4.1.1	Xml ostacolo all'aggregazione dati	53
2.1.4.2	URI: uniform resource description identifier	54
2.1.4.3	Resource description framework	55
2.1.4.3.1	Modello e sintassi	56
2.1.4.3.2	Schema	58
2.1.4.4	Sparql: linguaggio di query su rdf	60
2.1.4.4.1	Pattern matching	61
2.1.4.4.2	Formato dell'output	62

Capitolo III

3	Il sistema Hackystat	63
3.1	Dati dei sensori	65
3.2	Astrazioni di tipo progetto	71
3.3	Servizi componenti	73
3.3.1	Daily Project Data	73
3.3.2	Software Project Telemetry	74
3.3.3	Software Project Portfolio	76

Capitolo IV

4 Hackistat Linked Sensor Data	78
4.1 Architettura di sistema	78
4.2. Dati Hackystat utilizzati	80
4.3 Rete Hackystat	83
4.3.1 Algoritmo utilizzato	84
4.4 Linked Sensor Data	85
4.4.1 Linked Data degli sviluppatori	86
4.4.1.1 Collegamento con risorse esterne	87
4.4.1.1.1 Il servizio FOAF-QDOS	88
4.4.1.1.2 Il servizio SIOCLog	88
4.4.1.1.3 Il servizio RDFOhloh	89
4.4.2 Linked data dei progetti	89
4.4.2.1 1 Collegamento con risorse esterne	90
4.4.3 Linked Data degli issue	90
4.4.3.1 1 Collegamento con risorse esterne	91
4.4.4 Linked Data di risorse a libero accesso	92
4.5 Vocabolario Hackystat	92

Capitolo V

5 Implementazione di LiSeD	96
5.1 Interazione col modulo	96
5.1.1 Utilizzo del browser	96
5.1.1.1 API per utenti registrati	98
5.1.1.2 API per utenti qualsiasi	102
5.1.1.3 Dettagli sulle risorse	103
5.1.1.3.1 Risorse di tipo utente	105
5.1.1.3.2 Risorse di tipo progetto	107
5.1.1.3.3 Risorse di tipo issue	110

5.1.1.3.4 Risorse di tipo network	112
5.1.1.3.5 Risorse ad accesso libero	112
5.1.1.3.6 Risorse di tipo dataset Hackystat	113
5.1.1.3.7 Risorse di tipo cache	115
5.1.2 Utilizzo di un client Java	115
5.1.3 Utilizzo della GUI	116
5.1.3.1 Pannello Utenti	118
5.1.3.2 Pannello Progetti	122
5.1.3.3 Pannello issue	123
5.1.3.4 Pannello network	125
5.2 Linee guida seguite nella pubblicazione dei Linked Data	127
5.3 Esperimenti	128

Capitolo VI

CONCLUSIONI E SVILUPPI FUTURI 130

Bibliografia 133

INTRODUZIONE

Il lavoro svolto in questa tesi mira a migliorare la piattaforma Open Source Hackstat sviluppata presso il Collaborative Software Development Laboratory dell'Università delle Hawaii, ed è stato finanziato da Google Inc. nell'ambito del programma Google Summer of Code 2009. Hackstat è una piattaforma per il monitoraggio della qualità del lavoro degli sviluppatori coinvolti in uno o più progetti Software, e della qualità dei progetti Software stessi. Le informazioni base su cui si fondano tutte le successive analisi, sono i dati rilevati da sensori installati su tool di sviluppo. La sua utilità consiste nel supportare l'individuazione precisa e tempestiva dei problemi incontrati durante lo sviluppo Software, che possono altrimenti rallentare il processo di produzione. In altri termini si supporta l'attività di monitoraggio comunemente attribuita ai Project Manager, e si fornisce agli sviluppatori un'utile verifica delle proprie capacità. Il progetto svolto in questa tesi auspica a risolvere problemi di altra natura quali i rallentamenti non al processo di produzione generico, ma al lavoro individuale di ogni singolo sviluppatore. Rallentamenti comunemente dovuti alla necessità di apprendere le modalità di utilizzo di nuovi tool o librerie, di risolvere nuove e sconosciute eccezioni, di effettuare scelte progettuali mai incontrate prima, di ricevere suggerimenti da colleghi competenti su un determinato dominio. Tramite tecnologie nuove quali i Linked Data, si tenta la risoluzione di tali comuni problemi, supportando contemporaneamente la crescita del Web of Data, il futuro del Web, e la lo sviluppo collaborativo di Software.

Il Capitolo I descrive lo stato del Web attuale e i suoi problemi di gestione della conoscenza. Illustra in cosa consiste lo sviluppo collaborativo del Software, la sua utilità, e le soluzioni attualmente applicate ai relativi problemi.

Nel Capitolo II è invece descritto il futuro del Web, il Web of Data e le sue tecnologie principali. In particolare si descriveranno i vantaggi dei Linked Data. Il Capitolo III approfondisce le caratteristiche di Hackstat, la piattaforma su cui le idee contenute nella tesi sono state applicate; mentre nel Capitolo IV vengono mostrate in dettaglio le scelte progettuali intraprese, gli algoritmi utilizzati e l'esito della sperimentazione. Infine si riassumono i risultati raggiunti elencando i miglioramenti da apportare in futuro.

CAPITOLO I

1 LO SVILUPPO SOFTWARE

Lo sviluppo Software è l'insieme di attività che portano alla creazione di prodotti software. Lo sviluppo software può includere la ricerca, modifica, riutilizzo, re-ingegnerizzazione, manutenzione o qualunque altra attività il cui risultato sia un prodotto software [1]. In particolare la prima fase nello sviluppo software può coinvolgere differenti ambiti come il marketing, l'ingegneria, ricerca e sviluppo, e il management in generale.

Esistono diversi approcci allo sviluppo software, caratterizzati da un approccio più o meno strutturato, in cui il software evolve in passi successivi. La maggior parte delle metodologie condivide alcune combinazioni dei seguenti passi di sviluppo:

- Ricerca di mercato
- Raggiungimento dei requisiti necessari alla soluzione di business proposta
- Analisi del problema
- Pianificazione di una soluzione software-based
- Implementazione (programmazione) del software
- Testing del software
- Pubblicazione e Diffusione
- Manutenzione e correzione bug

Spesso ci si riferisce collettivamente a tali fasi con l'espressione "ciclo di vita dello sviluppo software" (SDLC). Gli approcci allo sviluppo si differenziano per un diverso ordine in cui vengono affrontate le fasi elencate, oppure per la quantità di tempo e dedizione dedicata ad una parte di esse piuttosto che ad altre. Anche il livello di dettaglio della documentazione prodotta ad ogni fase può variare.

I passi possono essere eseguiti uno per volta (approccio "a cascata") oppure possono essere iterati (approccio "extreme"). Gli approcci più "extreme" sono caratterizzati anche dal testing continuo attraverso l'intero ciclo di vita dello sviluppo, in modo da avere sempre un prodotto funzionante e privo di bug. Gli approcci invece più orientati verso il modello "a cascata" (Fig. 1.1) tentano di prevedere la maggior parte dei rischi e sviluppare un piano dettagliato prima di cominciare la fase di implementazione. In tal modo evitano cambiamenti significativi nella progettazione e conseguenti modifiche al codice durante fasi successive del ciclo di vita. Esistono rilevanti vantaggi e svantaggi relativi alle varie metodologie, e l'approccio migliore spesso dipende dal tipo di problema la cui

soluzione costituisce lo scopo della creazione del software in questione. Se il problema è ben compreso ed è effettivamente possibile pianificare in modo ottimale una soluzione già dalle prime fasi di sviluppo, allora l'approccio migliore potrebbe essere quello orientato al modello "a cascata". Se invece il problema è sconosciuto al team di sviluppo che non ha mai affrontato un problema simile, e la struttura della soluzione software non riesce ad essere facilmente immaginata, allora un approccio incrementale più "extreme" potrebbe essere preferibile.

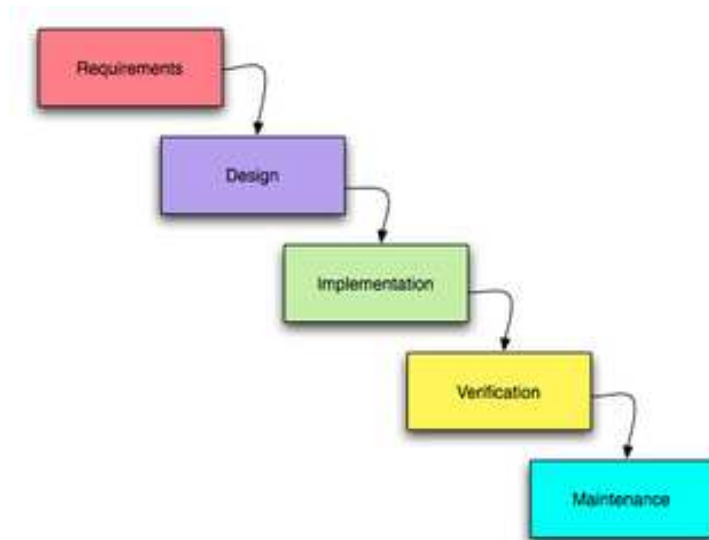


Fig. 1.1 – Attività del processo di sviluppo software rappresentate nel modello "a cascata".

Il processo di sviluppo software (anche detto "ciclo di vita del software" o "processo software") è una struttura che si impone sullo sviluppo di un prodotto software. Esistono diversi modelli per tali processi, ognuno dei quali descrive approcci per il compimento di una varietà di task incontrati durante il processo. Sono state eseguite indagini per individuare processi replicabili e prevedibili che migliorassero la produttività e la qualità. Alcuni di essi tentano di formalizzare la creazione del software, apparentemente priva di regole; altri applicano tecniche di gestione dei progetti [2], in quanto senza di esse i progetti software rischiano facilmente di non rispettare i tempi di sviluppo o di non rientrare nelle previsioni di spesa. Considerata però la quantità di progetti software che non soddisfano le aspettative in termini di funzionalità, costi o tempi, l'efficacia del project management pare essere fallace [3].

La creazione di un prodotto software può essere divisa nel Processo di Valutazione del Progetto (PEP) e nel Processo di Implementazione del Progetto (PIP). La fase PEP comprende l'attuazione di analisi dei requisiti e la pianificazione dell'architettura; può accadere che termini con la definizione di più di un unico progetto da implementare, allo scopo di limitare il range, i rischi e le risorse di gestione.

Generalmente i clienti hanno solo una vaga idea riguardo ciò che desiderano come prodotto finale, e nessuna idea riguardo ciò che il software dovrebbe permettere loro. Compito degli ingegneri del software più esperti è appunto l'individuazione di requisiti incompleti, ambigui o contraddittori, e il loro delineamento. Dopo aver rilevato i requisiti generici dal cliente, si determina un'analisi del range di sviluppo. Alcune funzionalità può accadere ricadano fuori dal range a causa dei limiti di costo o di requisiti mal definiti in fasi iniziale di sviluppo.

Il processo di Implementazione del Progetto include le seguenti fasi:

- Fase di Pianificazione del Progetto
- Fase di Design
- Fase di Build e Testing
- Fase di Test Integrati
- Fase di addestramento e Diffusione
- Fase di Chiusura

1.1 PIANIFICAZIONE DEL LAVORO

Lo scopo della fase di Pianificazione consiste nel definire il range e l'approccio, nel determinare le risorse necessarie e creare un piano di alto livello per il progetto. Questa fase include una transizione formale dal Processo di Valutazione al quello di Implementazione, e in essa si definisce anche la modalità di comunicazione e gestione di issue e cambiamenti di range.

Nella transizione dal PEP al PIP vengono trasferiti sia i documenti che la conoscenza creata da un Information Service Consultant (ISC) durante il PEP. Questi documenti non definiscono solo gli obiettivi e le funzionalità che verranno affrontati come parte del progetto, ma sono anche utili per la comprensione dettagliata dei requisiti funzionali per questo progetto tra team di sviluppo informatico, venditore e utenti finali. La documentazione che passa dal consulente ISC al Project Manager durante incontri successivi e scambi di informazioni, è di seguito elencata:

- Project Executive Summary Form compilato dal cliente, include una descrizione dei benefici attesi, effetti misurabili e stima dei costi.
- Definizione/Analisi del problema, Soluzione proposta e Analisi del Ritorno di Investimento (ROI). Il consulente ISC crea questi documenti in cui il problema di business è ben definito ed è analizzato il suo impatto operativo. Vengono anche identificate soluzioni alternative al problema, e tra esse la soluzione raccomandata, accompagnata da chiare e valide motivazioni. L'analisi ROI confronta il costo stimato per l'implementazione della soluzione con i benefici e/o i costi derivanti dalla mancata implementazione della proposta.

- Definizione del Range del Progetto creata durante il Processo di Valutazione dal consulente ISC, in cui si specificano le caratteristiche e funzionalità che dovranno essere incluse nel progetto. Vengono stabilite le scadenze temporali da rispettare nelle fasi successive e i limiti temporali di ogni fase.
- Architettura. La documentazione dettagliata deve identificare requisiti tecnici, interfaccia e impatto dei sistemi, vincoli di design, e l'approccio proposto per sviluppare la soluzione.
- Documento dei Requisiti anche detto Specifica Funzionale Dettagliata, sviluppato durante il PEP. Questo documento elenca tutte le risorse, i materiali e le caratteristiche necessarie per il progetto, e deve essere il più specifico e dettagliato possibile, in quanto pone le basi per il successo dell'implementazione. E' preferibile conservare traccia di tutte le proposte fatte dal venditore e le comunicazioni avvenute.
- Contratto. Una copia di tutti gli accordi finali riguardo prodotti e servizi per il progetto.
- Schedulazione della produzione. Base del Piano di Implementazione, fornisce una schedulazione dei requisiti e delle funzionalità che devono essere prodotte come parte del progetto, e identifica le parti responsabili di ognuna delle suddette fasi di produzione.

Il passaggio successivo consiste nel Project Kick-off che segna l'inizio dell'implementazione. In tale fase è inclusa la definizione dello scopo del progetto in un Project Overview utilizzando i documenti creati durante il PEP. Questo documento fornisce direttive non solo al team di sviluppo e ai venditori, ma anche a qualsiasi gruppo di lavoro che si crei eventualmente in futuro per completare il raggiungimento degli obiettivi di progetto. Il Project Overview Document comprende le seguenti informazioni:

- Executive summary. Definizione chiara della natura e degli obiettivi del progetto.
- Approccio all'Implementazione. Overview sulle caratteristiche che dovrebbe avere l'approccio all'implementazione. Per esempio un approccio multitasking potrebbe essere preferibile per ridurre il percorso critico del progetto. La scelta è spesso basata sull'architettura del sistema identificata nella fase di valutazione.
- Fase di implementazione e milestone. Identificazione di scadenze temporali per la fase di alto livello del progetto e per il completamento di ogni milestone prevista.
- Obiettivi di progetto e misurazioni. Gli obiettivi specifici del progetto devono essere dichiarati con chiarezza e le misurazioni devono essere identificate per ogni obiettivo. Le misurazioni determinano da quali fattori dipende il successo dell'implementazione, e tali fattori devono essere misurabili.

- Requisiti sulle risorse. Elenco di risorse necessarie al progetto, che include staff interno, servizi di consulenza, hardware, software, cavi, spazio, etc. In aggiunta alle risorse necessarie all'implementazione del progetto, dev'essere identificata chiaramente anche qualsiasi risorsa necessaria alla gestione e manutenzione del progetto.
- Analisi del rischio e piani di contingenza. Analisi dei vari rischi associati al progetto, e piani di contingenza per ogni fattore rischio significativo.

Successivamente viene organizzato il Project Team e il Gruppo di Lavoro, definendo le risorse umane necessarie e i ruoli e le responsabilità di ogni partecipante al progetto. Il ruolo del Project team consiste nel definire e approvare il piano di progetto dettagliato ed eseguire i compiti necessari al raggiungimento degli obiettivi prefissati. Ogni membro del team partecipa a tutti gli incontri schedulati, allo scopo di fornire aggiornamenti continui sullo stato del proprio lavoro, agli altri membri. Generalmente ogni membro rappresenta un'area dell'organizzazione.

Il ruolo del Gruppo di Lavoro invece, consiste nell'eseguire i compiti più specializzati necessari a completare il progetto. In altri termini i loro obiettivi sono specifici (per esempio addestramento dell'utente, test integrato, etc.). Ogni membro è tenuto a partecipare a tutti gli incontri schedulati. Il Leader del Gruppo di Lavoro si occupa di organizzare gli incontri del suo gruppo e comunicare al Project Team eventuali issue, cambiamenti di range del progetto, o semplicemente aggiornamenti sull'avanzamento del lavoro del proprio gruppo. Esempi di gruppi di lavoro possibili sono gruppi di testing (coordinano test integrati sulla quality assurance del progetto); di addestramento; tecnici (si interessano del design, compilazione e unit test sulla soluzione software); flusso di lavoro (definiscono i cambiamenti al flusso di lavoro, richiesti dal progetto).

Il task successivo riguarda la gestione degli issue e di cambiamenti al range del progetto. Si stabilisce la modalità di comunicazione e soluzione di qualsiasi issue o richieste di cambiamento di range che possono emergere durante lo sviluppo. Tali eventi necessitano infatti di essere comunicati al Project Manager e riportati nella documentazione regolarmente passata al responsabile del PEP. Un issue è un problema che emerge la cui soluzione non sembra essere banale, oppure c'è disaccordo riguardo quale soluzione adottare. Emerge all'interno del range del progetto in seguito ad azioni che possono partire da chiunque sia coinvolto nel progetto, e necessita di essere indagato e risolto. Si individua uno sviluppatore a cui viene assegnato il compito di risolvere l'issue, rispettando un'eventuale scadenza temporale.

Un cambiamento di range avviene nel momento in cui viene richiesta una modifica o miglioramento nelle risorse necessarie al progetto che esula dal range del progetto inizialmente definito nel Project Overview Document. L'approvazione o meno delle richieste è effettuata dal responsabile del PEP,

cui esse vengono inoltrate. In caso di approvazione viene definito l'impatto che si prevede avranno le modifiche sul piano delineato in precedenza, e può accadere che venga modificato anch'esso; in caso contrario la richiesta viene inclusa nella documentazione come possibile miglioramento futuro. Segue la fase di Project Design, i cui scopi e caratteristiche verranno analizzati nel paragrafo che segue.

1.1.1 PROJECT DESIGN

Lo scopo della fase di Project Design consiste nel delineare la soluzione software sia da un punto di vista tecnico che operativo, che soddisfi i requisiti del progetto identificati nel Project Overview Document. Da un punto di vista tecnico il design di un sistema include caratteristiche specifiche in linea con le dettagliate specifiche tecniche riguardanti sia l'applicazione che la costruzione dell'interfaccia. Da un punto di vista operativo invece, il processo di design include modifiche o aggiunte ad una qualunque delle attuali procedure operative che saranno necessarie per utilizzare il sistema ultimato. Il Documento di Design del Sistema deve includere le seguenti porzioni:

- Design dell'applicazione, comprende funzioni, caratteristiche, reportistica, distribuzione della reportistica, aspetti di sicurezza dell'applicazione. Ciò deve essere descritto in maniera dettagliata utilizzando diagrammi, prototipi e/o mockup.
- Design del database. Modifiche o aggiunte apportate al design della base di dati allo scopo di supportare il sistema. Comprende la specifica del tipo di base di dati e la sua posizione.
- Design dell'interfaccia e della comunicazione dati. Il design dell'interfaccia include tutti i sistemi che necessitano di essere interfacciati tra loro e il formato dei dati inviati e ricevuti da ognuno di tali sistemi.
- Design finale dell'architettura di sistema. Una rivisitazione dell'architettura è eseguita nel PEP. Tuttavia può accadere che l'architettura di sistema subisca modifiche con il procedere dello sviluppo, fino ad arrivare al design finale di questa fase.
- Piano di Unit Test, per testare l'applicazione sviluppata. Include scenari di Unit test, risultati dei test e loro assegnamento. Generalmente non comprende il testing delle interfacce, il quale rientra invece nella fase di testing integrato sulla Qualità Assurance.

Terminato il primo dei punti elencati ovvero il design dell'applicazione, esso viene presentato al Project Team per l'approvazione, in seguito alla quale il gruppo di lavoro comincia ad occuparsi dei successivi punti elencati. Ha inizio dunque il successivo task: la creazione del design di processo, avente lo scopo di definire le procedure di processo necessarie nei diversi domini di interesse per la

soluzione software proposta. Il design del processo include la documentazione e/o revisione delle procedure operazionali esistenti, revisione o creazione di nuove procedure operazionali necessarie per l'utilizzo della soluzione proposta, documentando qualunque criterio di utilizzo e procedura richiesta. Il documento di design del processo ultimato deve in seguito essere approvato dal Project Team. Il Project Team si occupa dunque di creare un piano dettagliato per il progetto, comprendente task dettagliati per ognuna delle restanti fasi dell'implementazione, e viene ripetutamente revisionato al termine di ognuna di tali fasi.

1.1.2 BUILD E UNIT TEST

Lo scopo della fase di Build e Unit Test consiste nella costruzione della soluzione software e nel testare le applicazioni costruite così come il design del processo. Durante questa fase l'hardware, il software, la rete, e tutte le risorse elencate come necessarie, vengono raccolte per l'utilizzo. Il gruppo di lavoro tecnico revisiona il design di sistema creato nella fase precedente, allo scopo di predisporre la costruzione del sistema. Tutta la documentazione di design del sistema e il Project Overview Document vengono presentati e discussi con tutti i membri del gruppo, per facilitare la comprensione del progetto e del sistema.

Nel caso sia necessario l'utilizzo di dati provenienti da altri sistemi pre-esistenti, un gruppo di lavoro dedicato si occupa di convertirli nel nuovo sistema. Il task di conversione dati include l'analisi dei dati attuali, la creazione di un piano di conversione che comprenda l'individuazione di quali dati devono essere convertiti; infine l'esecuzione del piano.

Segue la costruzione del sistema seguendo le direttive del documento di design. Tale fase comprende lo sviluppo di procedure tecniche associate alle componenti del sistema coinvolte (applicazioni, basi di dati, interfacce), quali ad esempio:

- Procedure di backup,
- Procedure di manutenzione del sistema,
- Procedure di Disaster Recovery,
- Procedure di controllo dei cambiamenti,
- Procedure di assistenza Help Desk.

Successivamente il gruppo di lavoro tecnico si dedica alla fase di esecuzione del piano di Unit Test creato durante la fase di design del sistema, per testarne ogni componente. Ciò comprende la verifica sia di tutti i possibili input che di tutti i possibili output di ogni componente; mentre sono esclusi i test tra sistemi (test integrati effettuati solo successivamente).

Dopo l'installazione e la configurazione del sistema effettuata dal gruppo di lavoro tecnico, comincia il task di verifica del design di processo assegnata ad un gruppo di lavoro dedicato. Lo scopo consiste nell'istruire il venditore della soluzione software riguardo al nuovo sistema, il quale viene appropriatamente configurato. Il design di processo creato viene dunque revisionato apportando eventuali opportune modifiche che dovranno essere approvate dal Project Team. Infine il piano per il progetto viene sottoposto alla revisione finale, in base alle informazioni raccolte fino a questo stadio del ciclo di vita del software.

1.1.3 TEST INTEGRATI SULLA QA

Lo scopo della fase di test integrati sulla Qualità Assurance (QA) comprende il testing su tutto il sistema incluse le interfacce. Esistono diversi tipi di testing quali unit, di funzionalità e integrati, da scegliere a seconda del contesto. A questo punto del ciclo di vita del progetto i requisiti e il design di sistema devono essere congelati per permettere lo svolgimento efficiente dei test integrati.

Qualsiasi modifica ai requisiti o al design richiede la riapplicazione di tutti i test.

Innanzitutto viene creato un Piano di QA test che comprende un elenco di tutti i requisiti dei test, ovvero di cosa è necessario testare, e che deve essere approvato dal Project team e quindi eseguito da un gruppo di lavoro dedicato. Il documento di Test Requirements è frutto della consultazione di tutta la documentazione sul progetto e del design di processo, ed è composto da un elenco categorizzato in base alle aree operazionali in cui sono descritte le sezioni da testare.

Successivamente viene creato un piano di QA test per definire step-by-step la modalità con cui ogni requisito del documento di Test Requirements dovrà essere testato. Ciò implica la descrizione di tutti gli scenari di test così come la designazione di ognuno di essi ai membri del gruppo di lavoro e la definizione di scadenze temporali. Uno scenario di test può essere realizzato tramite test manuale o automatizzato (utilizzando Testing tool). L'esecuzione del test integrato avviene secondo le direttive definite nel piano di QA test. I risultati vengono presentati al Project Team per l'approvazione o la definizione di ulteriori test. Infine il Project team revisiona nuovamente il piano di progetto.

Le successive fasi riguardano la pianificazione di un eventuale addestramento degli utenti finali e la diffusione del sistema software, cui segue la relativa manutenzione che può rendersi necessaria anche a distanza di molto tempo.

1.2 RALLENTAMENTI NEL CICLO DI SVILUPPO

Nel ciclo di vita del software si ripete spesso la definizione e revisione della timeline in quanto le scadenze temporali sono tanto importanti da rispettare quanto facilmente infrante. I rallentamenti più comuni e più influenti possono interessare allo stesso modo il Project Team nel delineamento di un piano di progetto o l'attività dei membri del gruppo di lavoro, in quanto entrambi spesso si trovano ad affrontare situazioni e problemi nuovi. Riassumendo i problemi più comunemente incontrati, anche all'interno di progetti di dimensioni ridotte, coinvolgenti un ristretto numero di sviluppatori, sono i seguenti:

1. Eccezioni a run-time;
2. Scelte implementative. Dubbi riguardo le scelte da prendere in fase di progettazione a causa di obiettivi e/o domini mai affrontati in precedenza;
3. Utilizzo di tool o librerie. Il riutilizzo del software pre-esistente e quindi di tool o librerie, è utile a ridurre il tempo necessario allo sviluppo. Tuttavia esiste un costo derivante dal tempo impiegato ad apprendere le modalità di un loro corretto utilizzo, che può essere più o meno elevato a seconda della complessità dei tool o della capacità di apprendimento degli sviluppatori.

Analizziamo singolarmente tali situazioni e le soluzioni che attualmente adottate per affrontarle, i cui limiti portano al rallentamento del processo di sviluppo.

1.2.1 ISSUE

Nel momento in cui uno sviluppatore, membro di un gruppo di lavoro o singolo, incontra un issue, cioè un problema la cui soluzione non è banale (che possa essere un'eccezione a run-time o altro) cerca informazioni provenienti da colleghi che hanno dovuto affrontare lo stesso problema in precedenza. In altre parole le soluzioni adottate sono le seguenti:

1. Forum e/o mailing list
 - a. Ricerca tramite comuni search engine di post pre-esistenti in cui si faccia riferimento all'issue di interesse. Lettura delle risposte fornite ai post individuati ed estrapolazione da essi di consigli e possibili soluzioni;

- b. Creazione di nuovi post in forum o mailing list, per esporre il proprio problema e chiedere aiuto alla comunità. Attendere una risposta alle proprie richieste dagli altri utenti.
- 2. Conversazione con colleghi sviluppatori per chiedere direttamente aiuto e suggerimenti
 - a. Tramite e-mail
 - b. Tramite instant-messaging

I limiti di tali soluzioni derivano dalla necessità di

- 1. Analizzare il linguaggio naturale. Considerata la forma non strutturata dei documenti web che comprendono il contenuto di forum e mailing list, ciò implica un alto costo in termini di tempo sia in forma di processo manuale e sia in forma automatizzata. In tal modo si filtrano i risultati di ricerca proposti dai comuni search engine, che spesso sono inefficaci in quanto basati su parole chiave e non concetti/relazioni. Non interpretano il linguaggio naturale il quale deve essere quindi interpretato manualmente in seguito.
- 2. Avere fiducia dei suggerimenti forniti dai colleghi contattati direttamente, spesso senza conoscerne l'effettiva competenza sul problema di interesse.

Infine in entrambi i casi è necessario attendere una risposta alle proprie richieste di aiuto, la quale non è immediata soprattutto se per la comunicazione si utilizzano mezzi quali forum, e-mail o mailing list.

1.2.2 SCELTE IMPLEMENTATIVE

Il termine “scelte implementative” è generico e comprende qualsiasi decisione debba essere assunta durante la lunga fase dell'implementazione. Come spiegato, generalmente le decisioni spettano ai membri del Project Team, e possono ad esempio consistere nell'approvazione o meno di un piano di test, di un algoritmo etc. Nel caso in cui il contesto e/o il dominio di applicazione di un progetto siano del tutto nuovi per i membri del team, assumere una decisione diventa particolarmente difficile. Anche in questo caso la soluzione più spesso adottata è la ricerca di esempi da parte di altri colleghi e progetti in cui si siano dovute effettuare le stesse scelte. Le azioni intraprese sono quindi le seguenti:

- 1. Progetti software altrui caratterizzati dallo stesso dominio di applicazione e/o dagli stessi obiettivi
 - a. Ricerca, tramite comuni search engine, nella documentazione di riferimenti alle scelte effettuate

- b. Ricerca, tramite open source code search engine, nel codice e nei commenti, di riferimenti alle scelte effettuate
- 2. Conversazione con colleghi sviluppatori per chiedere direttamente aiuto e suggerimenti
 - a. Tramite e-mail
 - b. Tramite instant-messaging

I limiti di tali soluzioni derivano dalla necessità di

1. Analizzare il linguaggio naturale. Considerata la forma non strutturata dei documenti web tra cui la documentazione dei progetti e il codice. Codice e documentazione sono gli unici strumenti grazie a cui è possibile rilevare dominio di interesse e obiettivi, e l'estrapolazione di tale informazione dal loro interno allo stesso tempo può risultare difficoltosa, a seconda che le varie sezioni siano accessibili tramite un indice o meno, che le informazioni più rilevanti siano evidenziate o meno, e a seconda della capacità dei loro autori di strutturare il discorso in linguaggio naturale al meglio. Tuttavia in generale la ricerca di informazione da un documento interamente non strutturato e in linguaggio naturale implica un alto costo in termini di tempo sia in caso tale processo venga effettuato manualmente e sia che venga automatizzato.
2. Il contenuto della documentazione dei progetti non è strutturato in campi che identifichino univocamente la semantica di ogni sua sezione, e i comuni search engine non sono in grado di effettuare ricerche su specifici campi semantici, bensì solo su parole chiave che possono essere fuorvianti, e non su concetti/relazioni. Di conseguenza può accadere che i risultati ricerca comprendano documentazioni di progetti che effettivamente non condividono dominio e obiettivi col progetto di interesse.
3. Anche gli Open Source code search engine non sono in grado di rilevare specifiche sezioni semantiche nel contenuto del codice o della documentazione dei progetti. Il più articolato è Codase (ancora in versione beta) che può effettuare ricerche specificatamente su chiamate/definizioni di metodi, definizioni di classi o loro attributi, variabili o riferimenti ad attributi di classe. I risultati di ricerca di Codase come vedremo sono i più affidabili. Tuttavia anche in questo caso non è possibile individuare con precisione le sezioni in cui si faccia riferimento a dominio e obiettivi: è necessario dedurli manualmente dai commenti nel codice o da descrizioni generiche, se esistono.
4. Avere fiducia dei suggerimenti forniti dai colleghi contattati direttamente, spesso senza conoscerne l'effettiva competenza riguardo al dominio e agli obiettivi di interesse. Inoltre è necessario attendere una risposta dai colleghi contattati direttamente, che può essere più o

meno immediata a seconda della disponibilità dell'interlocutore oltre che della modalità scelta per instaurare il contatto (e-mail o instant messaging).

1.2.3 UTILIZZO DI TOOL E LIBRERIE

La necessità di utilizzare nuovi tool o librerie comporta un costo dovuto all'apprendimento delle modalità di utilizzo. Non sempre viene fornita una chiara documentazione o degli esempi di utilizzo magari proprio nel linguaggio cercato. Le difficoltà di apprendimento possono causare non solo un rallentamento nel processo di sviluppo del software ma anche un abbassamento della qualità del software, perlomeno per quanto riguarda le sezioni di esso che utilizzino tali librerie in precedenza sconosciute. In tali situazioni si adottano le seguenti strategie:

1. Tutorial
 - a. Ricerca di tutorial o esempi relativi al tool o alla libreria di interesse
2. Progetti software altrui caratterizzati dall'utilizzo dei tool o delle librerie di interesse
 - a. Ricerca, tramite comuni search engine, di riferimenti a tool o librerie di interesse all'interno della documentazione
 - b. Ricerca, tramite open source code search engine, nel codice e nei commenti, di riferimenti a tool o librerie di interesse
3. Conversazione con colleghi sviluppatori per chiedere direttamente aiuto e suggerimenti
 - a. Tramite e-mail
 - b. Tramite instant-messaging

Le soluzioni comunemente adottate anche in questo caso presentano limiti relativi al costo necessario ad analizzare il linguaggio naturale di documenti quali tutorial, documentazione o codice; limiti dei comuni search engine e degli OS code search engine; assenza di garanzie sull'affidabilità di colleghi consultati direttamente, e necessità di attendere (per una quantità di tempo imprevedibile) una loro risposta.

A dimostrazione della veridicità dei problemi, soluzioni e limiti individuati nelle ultime tre sezioni, segue la descrizione dei passi realmente effettuati per risolvere lo scenario riguardo la difficoltà nell'utilizzo e individuazione di nuove librerie .

1.2.4 SCENARIO

Si è scelto di verificare quanto effettivamente nella realtà i limiti relativi alle soluzioni ai problemi elencati, comuni durante lo sviluppo del software, possa rallentare il ciclo di vita e presentare spesso insoddisfacenti soluzioni finali.

Supponiamo che ad un Gruppo di Lavoro venga presentato come problema per cui creare una soluzione software, il filtraggio delle e-mail, e che il linguaggio principale da utilizzare (in quanto utilizzato già in altre componenti di un pre-esistente sistema da estendere) sia Java. Infine supponiamo che nessun membro del gruppo abbia mai creato filtri per e-mail, tanto meno utilizzando Java, e di conseguenza si renda necessaria la ricerca delle migliori librerie che possano supportare l'obiettivo assegnato loro. In altre parole il gruppo deve affrontare difficoltà nell'utilizzo e anche nella sola individuazione di nuove librerie.

Il gruppo quindi nell'ordine decide di

1. cercare progetti software in cui si sia affrontato lo stesso problema, in Java, allo scopo di trovare quali librerie sono state utilizzate e allo stesso tempo esempi di utilizzo.
2. cercare tutorial online
3. contattare direttamente colleghi sviluppatori per ottenere dei suggerimenti.

Per attuare la prima strategia vengono utilizzati i più diffusi OS code search engine, ovvero: Codase, Google code search, Koders.

1.2.4.1 RICERCA TRAMITE CODASE

Codase permette di effettuare ricerche specifiche su chiamate o definizioni di metodi, attributi o definizioni di classi, variabili e riferimenti ad attributi. Nessuna di queste opzioni è utile nel caso attuale e quindi viene effettuata una ricerca generica con query "e-mail filter" e linguaggio di programmazione "Java" (Fig. 1.2).



Fig. 1.2 – Ricerca generica su Codase

In seguito all'esecuzione di tale ricerca si ottengono numerosi risultati tra cui verranno considerati solo i primi 5 per semplicità e anche perché in una situazione reale non c'è tempo di esaminarli tutti e ci si limita ai primi risultati o comunque alla prima pagina di risultati. I primi 5 risultati sono i progetti "LDAP account sync", "JFetch", "ebXML-based mail client", "J", "XToken", ognuno correlato con le porzioni di codice il cui contenuto nelle sue parti evidenziate, ha causato l'inserimento del progetto tra i risultati ricerca, per fornire una giustificazione all'utente (Fig. 1.3). Grazie a tale giustificazione è subito chiaro che il primo risultato relativo al progetto "LDAP account sync" non è di interesse, in quanto come giustificazione viene presentato codice html che casualmente contiene le parole chiave cercate. Si investiga quindi sul secondo risultato: "JFetch" (Fig. 1.3).

2. [SenderMailFilter.java](#) [Project: **JFetch**] [Language: java] [LOC: 145]

```
try {
    message.setFlag(Flags.Flag.DELETED, true);
} catch(MessagingException me) {
    logger.error("Could not mark message for deletion", me);
}
} else {
    logger.info("Message sender found in list, skipping, " +
        MessageUtil.toString(message));
}
```

Fig. 1.3 – Secondo dei risultati ottenuti per la ricerca effettuata da Codase, corredato dalla porzione di codice che maggiormente giustifica il suo inserimento nei risultati.

Il progetto è descritto su un sito web (Fig. 1.4) e dalla descrizione emerge che il progetto in questione contiene potenzialmente informazioni utili riguardo il filtraggio di e-mail in Java. Purtroppo però non è disponibile alcuna documentazione e, quindi, si tenta di visualizzare i file contenuti nell'implementazione del progetto ipotizzando che siano visualizzabili seguendo il link fornito per il download .




Fig. 1.4 – Menu principale del sito web che descrive il progetto JFetch

Il link per il Download porta ad una pagina su Sourceforge in cui, pur cliccando sul bottone "View all files" non vengono visualizzati tutti i file .java eventualmente consultabili, bensì solo gli archivi di binari e sorgenti scaricabili (Fig. 1.5). Di conseguenza è necessario scaricare l'intero archivio di

codici sorgente e identificare le librerie utilizzate, eventualmente con l'aiuto di un IDE all'interno del quale importare il progetto. Dopo averle individuate è necessario determinare le porzioni di codice in cui vengono utilizzate per trovare esempi di utilizzo: nel caso in cui si utilizzino sistemi operativi Linux-based ciò si riduce all'utilizzo di un opportuno comando bash di parsing del testo dei file .java; altrimenti sarà necessario aprire manualmente i file e cercare al loro interno. In entrambi i casi l'iter necessario a scoprire le librerie utilizzate da JFetch per il filtraggio delle e-mail e ad individuare all'interno del codice degli esempi di utilizzo, è estremamente lungo e dispendioso in termini di tempo.

SourceForge.net > Find Software > JFetch > Browse Files


JFetch by [execve](#)

[Summary](#)
[Files](#)
[Support](#)
[Develop](#)

JFetch is an application to download your email through protocols like POP3 and IMAP. The decision of whether to download a mail or not is made through a sequence of filters. The user can easily add customized filters to this sequence.

[Download Now!](#)
 JFetch-1.1a-src.tar.gz (1.5 MB)
 OR
 [View all files](#)

Browse Files for JFetch

File/Folder Name	Platform	Size	Date ↓	Downloads	Notes/Subscribe
Newest Files					
JFetch-1.1a-src.tar.gz		1.5 MB	2002-10-16	747	
JFetch-1.1a-bin.tar.gz		1.2 MB	2002-10-16	694	
All Files					
▼ jfetch		7.6 MB	2002-10-16	3,483	
▼ Jfetch-1.1a		2.6 MB	2002-10-16	1,441	
JFetch-1.1a-src.tar.gz		1.5 MB	2002-10-16	747	
JFetch-1.1a-bin.tar.gz		1.2 MB	2002-10-16	694	
▼ jfetch-1.0		2.6 MB	2001-04-07	1,001	

Fig. 1.5 – Pagina Sourceforge che ospita i file scaricabili del progetto JFetch.

Il terzo risultato della ricerca effettuata tramite Codase è invece il progetto “ebXML-based mail client”. In tal caso le descrizioni del progetto disponibili sono una pagina su Sourceforge contenente una descrizione ridotta e generica da cui non è possibile affermare con certezza l'utilità o meno del

progetto ai nostri scopi; e una pagina su freebXML, un repository che ospita progetti software a supporto dello sviluppo e dell'adozione di ebXML e delle relative tecnologie.

Project: ebMail

- Download
- Mailing Lists
- Product Sheets

ebMail is a GUI system which helps users with minimal knowledge on ebXML to engage in B2B activities. The user interface is email-client-like to lower the learning barrier. Underlying, the system makes use of open standards (ebXML) to communicate with business partners. Since the business messages are composed and read in GUI form, this project is useful to SMEs who do not need backend integration. Business process flow can be guided by the plugins of ebMail. ebMail supports grouping of business messages into threads.

The project is developed using Java, therefore it is platform-neutral. The GUI part is using Java Swing. For the ebXML Messaging Service, ebMail makes use of another open source project **Hermes**. Currently, ebXML packaging and digital signature are supported, and the ebXML transport protocol is binded to SMTP.

Fig. 1.6 – Pagina FreebXML che ospita il progetto ebXML-based mail client.

Da questa pagina (Fig. 1.6) è possibile visualizzare una brochure che non si rivela essere di interesse in quanto presentazione destinata all'utilizzatore finale in cui non sono descritti i dettagli dell'implementazione. Si tenta dunque di trovare maggiori informazioni nella pagina del Download (Fig. 1.7) che effettivamente si rivela contenere collegamenti ad una buona quantità di documentazione oltre che al codice sorgente. Tuttavia tutte le risorse collegate possono essere visualizzate solo previo inserimento di alcuni dati personali in un form che non risulta essere funzionante: dopo aver inserito i dati e cliccato sull'unico bottone "next" disponibile oltre a "reset", si viene riportati sullo stesso form vuoto, e all'indirizzo e-mail appena fornito non perviene alcuna e-mail di notifica o altro.

Download - ebMail (release 1.2)

Install shield package for Win32 (To be released)

This is a Win32 installation setup file that installs the binaries of ebMail on a Win32 system. This package will install a sample plug-in. Also, JRE is included.

Setup Guide

This document describes the procedures to setup ebMail and import plugins.

Plugin Development Guide

This document describes the architecture for plugin of ebMail. It provides information on how to develop plugins for ebMail.

Plugin Development Tutorial

This document presents an example of developing a plugin for ebMail.

Source Distribution

This is the source package of ebMail. It is identical to the source code that can be downloaded from SourceForge. The source code of a sample plugin is included.

Binary Distribution

This is a pre-compiled package of ebMail, which is executable without the need of compilation. A sample plugin is included.

Fig. 1.7 – Pagina FreebXML che ospita la sezione "Download" del progetto ebXML-based mail client.

Questo accade nonostante tra le FAQ del repository sia specificato che non occorre alcuna registrazione per l'accesso ai contenuti ospitati, i quali sono dunque tutti liberamente accessibili. Comunque anche se il form fosse stato funzionante sarebbe stato necessario impiegare del tempo a individuare le informazioni cercate riguardo le librerie, nel documento in linguaggio naturale "Plugin development guide" (Fig. 1.7) oppure all'interno del codice sorgente stesso. Il quarto progetto elencato nei risultati ricerca è "J". Il sito web che lo descrive non ha un menu principale ma, individuato il link alla guida utente, si visiona l'indice della guida. Nonostante il software J sia un editor di testo e quindi sembrerebbe non avere attinenza con il filtraggio e-mail cercato, nell'indice troviamo un capitolo intitolato "Mail". Tuttavia anche in tale capitolo non è contenuta la parola chiave "filter" (Fig. 1.8). Si decide di abbandonare l'indagine sul progetto, il cui codice sorgente sarebbe eventualmente stato scaricabile tramite Sourceforge.

Mail

J provides support for IMAP, POP and SMTP. (By coincidence, it's also possible to use the [openMailbox](#) command to open a read-only view of a Unix mbox-style mailbox, but that's not officially sanctioned yet.)

× Trova: filter ↓ Successivo ↑ Precedente 🔍 Evidenzia ☐ Maiuscole/minuscole ⓘ Testo non trovato

Fig. 1.8 – Capitolo intitolato "Mail" interno alla guida utente del progetto J. Una ricerca della parola chiave "filter" non produce alcun risultato.

Infine il quinto progetto nei risultati ricerca di Codase è "XToken". L'unico sito web di riferimento è però Sourceforge e quindi, vista la descrizione che lascia intuire la presenza di utili informazioni estraibili dal codice, si procede a scaricare l'archivio del codice sorgente e cercare le librerie usate e gli esempi di utilizzo all'interno del codice, manualmente. XToken viene descritto come un contatore di messaggi di SPAM che etichetta opportunamente.

I risultati ottenuti effettuando una ricerca tramite il code search engine Codase sono stati abbastanza soddisfacenti, ma la ricerca delle librerie utilizzate nel codice di interesse si è rivelata un processo dispendioso in termini di tempo.

1.2.4.2 RICERCA TRAMITE KODERS

Nel caso in cui si utilizzi il code search engine Koders per ricercare il codice sorgente di progetti software altrui tra i cui obiettivi sia compreso il filtraggio delle e-mail e che utilizzino Java, i risultati sono meno soddisfacenti rispetto a quelli ottenuti con Codase. Il vantaggio però consiste nel

non essere costretti a scaricare il sorgente per accedere ai singoli file dell'implementazione dei progetti: i file di tutto il progetto sono sfogliabili dall'interno di Koders stesso.

La query utilizzata per la ricerca è sempre “e-mail filter” e la prima pagina di risultati contiene 20 riferimenti a progetti, ad ognuno dei quali è associata una breve giustificazione del motivo per cui sono stati considerati rilevanti nella ricerca. Tra i primi 20 risultati, i primi 16 si riferiscono tutti ad un solo progetto: Homesuite. Homesuite è una suite di applicazioni web-based comprendente e-mail, rubrica di indirizzi, diario, segnalibri, etc. che permette di sincronizzare basi di dati diverse e condividere dati con altri utenti. Tale risultato si rivela utile: esplorando l'implementazione del progetto dall'interno stesso di Koders si individuano facilmente dal nome, i file che fanno riferimento al filtraggio e-mail (Fig. 1.9).

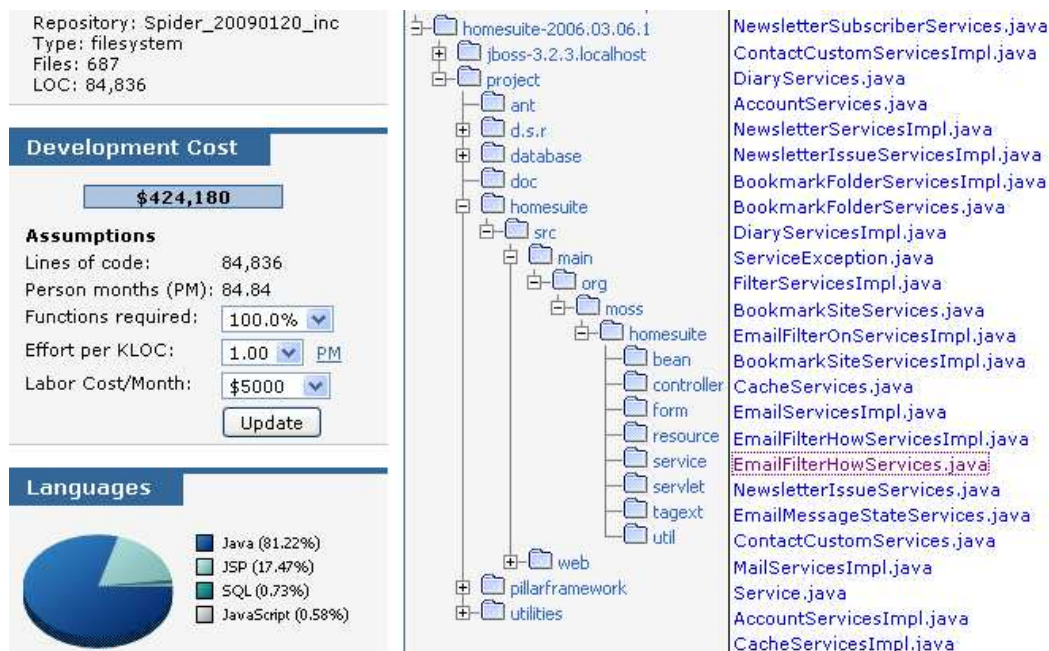


Fig. 1.9 – Interfaccia di esplorazione file per il progetto Homesuite, all'interno di Koders.

In 17° posizione tra i risultati vi è il progetto dotii, ovvero un multi-tool web-oriented comprendente un mailserver, webmail, webcalendar, filesaver e altro. Anche in questo caso si tratta di un progetto effettivamente rilevante per la ricerca e di cui è possibile la consultazione online immediata di tutti i file del progetto.

Il progetto che segue, “Utils”, non è invece particolarmente rilevante in quanto fornisce solo un'interfaccia per il filtraggio e-mail e nient'altro. Infine il progetto “Deep Email Miner” è di media rilevanza in quanto fornisce esempi specifici di analisi del contenuto delle e-mail approfondito, che può risultare utile al filtraggio anche se non vi sono esempi effettivi di filtraggio e-mail.

La consultazione dell'implementazione dei progetti è quindi molto più rapida e non comporta ulteriori costi per l'esplorazione di più o meno completi siti web dedicati ai progetti. Tuttavia resta l'alto costo per l'esplorazione manuale del codice.

1.2.4.3 RICERCA TRAMITE GOOGLE CODE SEARCH

La ricerca tramite l'engine di Google Code Search, utilizzando sempre la query "email filter" che precede l'espressione lang:java come richiesto, nonostante sia abbastanza soddisfacente presenta il maggior numero di svantaggi a causa della strutturazione dell'interfaccia di tale engine. I risultati fanno riferimento a file .java di progetti che effettivamente coinvolgono il filtraggio e-mail, ed è possibile esplorare i file dei progetti dall'interno di google code search evitando di dover scaricare l'intero archivio di codice sorgente.

Tuttavia i file dei progetti evidenziati esplicitamente nei risultati ricerca, non sono esattamente i file più rilevanti per la ricerca all'interno di quegli stessi progetti. I file più rilevanti vanno cercati manualmente, il che richiede alto costo in termini di tempo soprattutto per orientarsi all'interno di progetti molto grandi. Inoltre è assente qualsiasi descrizione, anche breve, dei progetti i cui i file ospitati su Google Code Search appartengono, a differenza di quanto avviene sia su Codase che su Kodors. Ciò rende meno immediata la comprensione della rilevanza o meno di un risultato ricerca, obbligando ad effettuare un'ulteriore ricerca su Internet del nome del progetto, e perfino l'intuizione del nome dal path dei file ospitati, in quanto il nome dei progetti non è in alcun modo esplicitato.

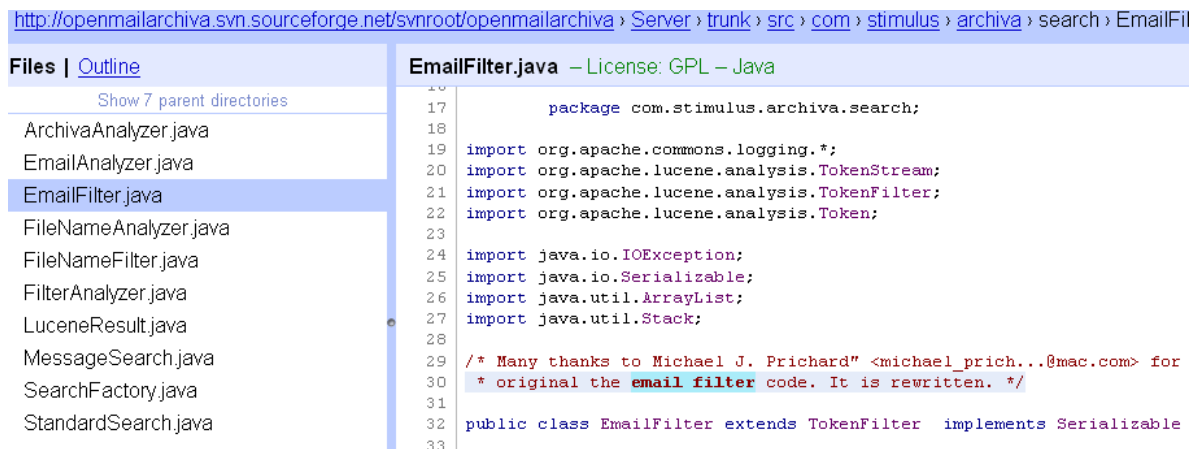
Tra i primi 5 risultati della ricerca effettuata vi sono rispettivamente file appartenenti ai progetti Apache Jakarta JMeter, Apache JSieve, Htmlanalyser, Circularparse, MailArchiva.

Il primo risultato, Apache JMeter, è un tool sviluppato per il monitoraggio di test e performance e non coinvolge alcun filtraggio di e-mail. Sembra sia stato inserito nei risultati solo perché vengono nominate le parole "e-mail filter" all'interno di codice html. Il secondo risultato è invece altamente rilevante. Si tratta infatti dell'implementazione in Java del linguaggio Sieve per il filtraggio delle mail definito dall'RFC3028. JSieve può essere integrato in un'applicazione di posta elettronica per aggiungervi supporto al linguaggio Sieve. Effettuando un'ulteriore ricerca tramite un comune search engine è possibile trovare la pagina web Apache completa di tutta la documentazione (in linguaggi naturale e quindi da esplorare manualmente).

Relativamente al terzo risultato, Htmlanalyser, non è visualizzabile né la pagina web ospitata su Google Code, né l'svn, in quanto l'accesso risulta limitato ai possessori di particolari permessi.

Il quarto risultato, Circularparse, è associato (previa ricerca tramite comune search engine) ad una pagina web ospitata su Google Code, priva di documentazione o mailing list. Non è chiaro l'obiettivo del progetto ed esplorandone i file sembra che non sia rilevante alla ricerca.

L'ultimo risultato è invece rilevante (Fig. 1.10), e contiene una classe che effettua esattamente il filtraggio e-mail (utilizzando librerie di Apache Lucene).



```
10
11
12
13
14
15
16
17     package com.stimulus.archiva.search;
18
19     import org.apache.commons.logging.*;
20     import org.apache.lucene.analysis.TokenStream;
21     import org.apache.lucene.analysis.TokenFilter;
22     import org.apache.lucene.analysis.Token;
23
24     import java.io.IOException;
25     import java.io.Serializable;
26     import java.util.ArrayList;
27     import java.util.Stack;
28
29     /* Many thanks to Michael J. Prichard" <michael_prich...@mac.com> for
30     * original the email filter code. It is rewritten. */
31
32     public class EmailFilter extends TokenFilter implements Serializable
33
```

Fig. 1.10 – Visualizzazione di un file del progetto MailArchive riferito al filtraggio e-mail, all'interno di Google Code Search.

In conclusione utilizzando Google Code Search si ottiene un maggior numero di risultati insoddisfacenti cui si aggiungono svantaggi legati all'usabilità dell'interfaccia presentata in cui sono assenti collegamenti ai progetti cui i file ospitati appartengono, descrizioni di tali progetti ed esplicitazione del loro nome.

1.2.4.4 RICERCA TRAMITE GOOGLE

In alternativa all'utilizzo di Code Search Engine, è possibile utilizzare search engine comuni. Nel nostro esempio è stato utilizzato Google in quanto è il più comunemente utilizzato e offre generalmente risultati migliori rispetto ad altri. La query inserita è stata "e-mail filter java library" e ha portato ai risultati visualizzati in Fig. 1.11. I motori di ricerca comuni raramente presentano tra i risultati di ricerca le pagine di progetti software open source. Molto più spesso vengono visualizzati invece riferimenti ad articoli e Javadoc ed API.



Fig. 1.11 – Risultati della ricerca effettuata sul motore di ricerca Google per la query “e-mail filter java library”.

Anche in questo caso infatti, i primi risultati per la ricerca effettuata sono nell’ordine: un articolo riguardante filtri su Tomcat 5, Google Data Protocol, JavaMail API, Javadoc del progetto RIM device, una libreria grafica, un elenco di software Java per il filtraggio e-mail e un articolo sulla libreria Classifier4J. I primi due risultati insieme alla libreria grafica e all’elenco di software (in cui l’unico software effettivamente riferito al filtraggio delle e-mail non è né free-ware né open source) sono irrilevanti. Gli unici risultati che possono essere sfruttati sono la JavaMail API, l’articolo riguardante la libreria Classifier4J che sembra poter tornare utile al filtraggio e-mail e la JavaDoc (Fig. 1.12). Tra queste neanche la JavaDoc, dopo un ulteriore approfondimento, si rivela utile, in quanto il filtraggio effettuato nella classe EmailAddressTextFilter è estremamente superficiale e non coinvolge particolari librerie.

net.rim.device.api.ui.text
Class TextFilter

[java.lang.Object](#)
|
+--[net.rim.device.api.ui.text.TextFilter](#)

Direct Known Subclasses:

[EmailAddressTextFilter](#), [HexadecimalTextFilter](#), [IPTextFilter](#), [LowercaseTextFilter](#), [NumericTextFilter](#),
[PhoneTextFilter](#), [UppercaseTextFilter](#), [URLTextFilter](#)

public abstract class **TextFilter**
extends [Object](#)

Fig. 1.12 – JavaDoc relativa alla classe TextFilter in cui si trova un utile riferimento alla classe EmailAddressTextFilter, entrambe appartenenti al progetto Rim Device.

I risultati utili si riducono a soli due su cinque contenuti nella prima pagina di risultati dunque, e in entrambi i casi sono necessarie ulteriori ricerche per approfondirne esempi di utilizzo e descrizione.

In conclusione le ricerche effettuate tramite comuni motori di ricerca sono meno soddisfacenti di quelle effettuate utilizzando motori di ricerca dedicati al codice di progetti open source, ma in entrambi i casi l'individuazione di ciò che si cerca, ad esempio di librerie utilizzate ed esempi di utilizzo, richiede un costo sempre mediamente alto in termini di tempo.

1.2.5 LIMITI DEI MOTORI DI RICERCA

Come abbiamo anticipato nell'introduzione, il Web può essere definito come una rete di risorse di informazioni, basata sull'infrastruttura di internet. Le pagine web sono collegate sintatticamente mediante indici che localizzano l' URL della pagina e tali collegamenti consentono di identificare le pagine in modo univoco. Uno dei principali limiti di tale impostazione risiede nell'assenza di significato dei collegamenti, in altre parole questo sistema manca di una qualche capacità semantica: i collegamenti dovrebbero non solo condurci in un determinato luogo (la pagina web) ma anche descriverci il luogo in cui saremmo condotti. Lo schema di collegamenti che abbiamo oggi a disposizione può essere rappresentato sostanzialmente con l'immagine in Fig. 1.13.

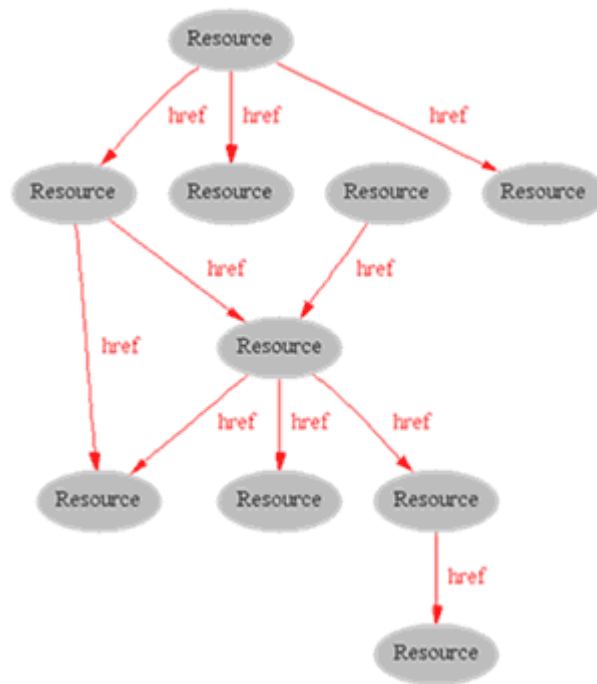


Fig. 1.13 – Collegamenti sintattici attualmente disponibili nel Web of Documents [14].

Uno degli strumenti più diffusi per interagire e cercare risorse informative sul web è rappresentato dai motori di ricerca. Il funzionamento base di un motore di ricerca, tralasciando le caratteristiche più sofisticate, può essere descritto nel seguente modo: l'interazione fra l'utente e il motore di ricerca inizia con l'invio di un'interrogazione, tramite form HTML; il motore di ricerca utilizza le parole dell'interrogazione per cercare nei file indice che si è precedentemente costruito scaricando e analizzando le pagine web, quali pagine contengono quelle parole; tali pagine vengono quindi ordinate per pertinenza utilizzando vari criteri, che essenzialmente si basano sul contenuto testuale delle pagine stesse e sulle informazioni rappresentate dai link sul web che puntano ad esse; il risultato viene mostrato all'utente utilizzando una pagina HTML che contiene rappresentazioni condensate delle pagine più pertinenti.

Pur fornendoci un servizio indispensabile, i motori di ricerca soffrono di evidenti limiti che si manifestano palesemente andando ad analizzare i risultati restituiti alle nostre ricerche. Il primo dato da mettere in evidenza a questo riguardo è l'esistenza del cosiddetto web nascosto (ovvero una quantità di risorse informative disponibili sul web ma non rintracciabili dai motori di ricerca per varie cause quali contenuti non indicizzati, pagine periferiche, immagini, files audio, files video, file flash, archivi zippati, informazioni contenute in basi di dati, contenuti dinamici che cambiano in tempo reale, ecc.) stimato essere pari all'80% delle risorse disponibili (anche se tali dati devono essere considerati con grande cautela, data l'impossibilità di una misurazione precisa). Ma questo non è certo l'unico problema: problemi di vocabolario; visualizzazione dei risultati poco intuitiva ed

esplicativa; assenza di risultati; bassa pertinenza con la richiesta inviata sono solo alcuni dei limiti di fronte a cui i motori di ricerca si trovano. In riferimento ai problemi di vocabolario possiamo esemplificare casi di sinonimia e polisemia che rendono praticamente impossibile per i motori di ricerca restituire esclusivamente i risultati da noi attesi, questo a causa della notevole ricchezza (ma anche ambiguità) del linguaggio naturale, di fronte a cui anche i sistemi di ricerca più evoluti soffrono di enormi limiti di interpretazione.

La realizzazione completa dell'architettura del web semantico consente di potenziare enormemente le capacità di ricerca dei motori attuali e, di conseguenza, di incrementare sensibilmente le potenzialità del web. Tuttavia anche i motori di ricerca semantici presentano dei limiti a causa dei quali non si sono mai diffusi su larga scala.

1.2.6 LIMITI DEI MOTORI DI RICERCA SEMANTICI

I motori di ricerca semantici offrono la possibilità di effettuare ricerche specifiche su ben definiti, non ambigui, concetti e relazioni; in quanto i dati manipolati da tali tipologie di motori sono metadata che, come vedremo, descrivono le risorse univocamente identificabili nel loro significato. Le ricerche quindi non sono più effettuate su documenti in linguaggio naturale da cui è possibile estrarre solo parole chiave, bensì sono effettuate su descrizioni strutturate, in cui ogni componente descritta ha un preciso, condiviso ed esplicito significato. Ciò semplifica anche la consultazione dei contenuti, in quanto non è più necessario l'analisi del linguaggio naturale manualmente: le informazioni sono più facilmente comprensibili dall'uomo ma sono anche machine-understandable.

Purtroppo però l'alto costo di creazione di tali metadata (che verrà approfondito in seguito) ha limitato la quantità di dati su cui tali motori possono operare. Inoltre ogni dataset spesso utilizza vocabolari differenti nelle descrizioni delle proprie risorse, il cui significato di conseguenza è spesso uguale ma associato ad identificatori univoci differenti [15]. Per esempio può accadere di effettuare una ricerca di dataset contenenti nelle proprie descrizioni la tripla <*, vocabolario_1:possiede, vocabolario_1:PC> e di non trovare tutti i dataset in cui effettivamente si afferma che una qualche risorsa possiede un PC a causa dell'utilizzo, in alcuni dataset di relazioni e concetti del tipo vocabolario_n:Possiede, oppure vocabolario_n:ha, e vocabolario_n:PC, oppure vocabolario_n:PersonalComputer; senza che venga esplicitata l'equivalenza semantica tra tali concetti e relazioni provenienti da differenti vocabolari.

Per concludere quindi, i maggiori limiti incontrati dai motori di ricerca semantici sono:

- uso nei dataset di identificatori universali diversi per uguali concetti
- mancanza di connessioni tra risorse simili
- Ridotta quantità di dataset di metadata online (causa: elevato costo di creazione).

Una conferma dell'inefficacia legata a tali cause è la diffusione dei soli motori semantici special-purpose, ovvero legati ad un unico particolare vocabolario: in questo modo viene meno il primo degli ostacoli elencati.

1.3 SOLUZIONE PROPOSTA: PROFILI COLLEGATI E DESCRIZIONI DI DATASET

Le difficoltà elencate che emergono comunemente durante il ciclo di sviluppo del software, presentano diverse soluzioni comunemente adottate ma tutte accomunate dalla ricerca della collaborazione altrui, del frutto delle esperienze accumulate in altri progetti relazionabili in qualche modo col proprio. Si tratta dunque, di uno sviluppo del software che diventa collaborativo grazie alla connessione di software diversi tramite lo strumento attuale più collaborativo esistente, ovvero il Web. Se si creassero dei profili per i progetti e gli issue e gli utenti, comprendenti espliciti collegamenti generati dinamicamente a tutte le altre esistenti descrizioni di progetti, issue e sviluppatori, il tempo necessario per individuare l'informazione cercata si ridurrebbe ad un semplice click sul link visualizzato. Se le descrizioni relative agli sviluppatori contenessero anche informazioni precise sulle sue effettive capacità, ciò potrebbe essere usato come garanzia di competenza nel momento in cui un altro sviluppatore necessita di contattare direttamente un collega, per chiedere consigli instaurando una conversazione. Le risorse e gli strumenti necessari per realizzare ciò sono:

- Meta-descrizioni di progetti, issue e utenti. La tecnologia utilizzabile è l'RDF.
- Piattaforma che raccolga spontaneamente informazioni relative a progetti, issue e utenti, pre-esistente; da cui creare le meta-descrizioni automaticamente. In particolare dovrebbe collezionare informazioni sulle capacità e competenze degli sviluppatori. La piattaforma individuata a tale scopo è Hackstat.

- Collegamenti creati dinamicamente ad altre meta-descrizioni relazionabili in base ad alcuni principi (ad esempio progetti aventi stesso obiettivo, che utilizzano stessi tool o stesse librerie), con quelle locali. La tecnologia utilizzabile sono i Linked Data, di recente diffusione.

Grazie a queste tecnologie, è possibile avere descrizioni di progetti in cui le informazioni relative alle librerie utilizzate, tool e obiettivi non debbano essere manualmente estrapolate da documentazione, se esistente, scritta in linguaggio naturale; bensì siano immediatamente individuabili in quanto contenute in descrizioni schematizzate, associate a campi dalla semantica esplicitata univocamente in modo non ambiguo.

Per quanto riguarda le ricerche specifiche su tool o librerie, o particolari competenze che si preferisce abbiano gli sviluppatori cui si desidera chiedere un suggerimento, è possibile utilizzare gli attuali motori di ricerca semantici seguendo una strategia in fase di standardizzazione [17], che permette di superarne i limiti descritti nel precedente paragrafo. Tale strategia consiste nell'associare al proprio dataset, tramite opportune specifiche all'interno di una sitemap estesa con la Semantic Web Crawling Sitemap Extension [18], una meta-descrizione del dataset stesso in cui si utilizzi il Vocabulary of Interlinked Datasets (voID) [16]. voID permette di specificare i concetti soggetto del dataset, per identificare i quali si è scelto di utilizzare Umbel [19]. Umbel è

- Upper Mapping and Binding exchange Layer
- Lightweight ontology

Il suo obiettivo consiste nel relazionare i dati del Web con un insieme standard di Subject Concept in modo da fornire un contesto utile a migliorare le descrizioni delle risorse. Il contesto fornito è particolarmente articolato e completo, se si considera che Umbel mappa 20.000 nodi/subject concepts solidamente interconnessi.

Per supportare la collaborazione e quindi la pubblicazione dei dati Hackystat in tal modo fruibili da utenti esterni, si è scelto di pubblicare SPARQL endpoint. In particolare voID permette di specificare nella descrizione del dataset:

- i soggetti del dataset
 - <http://purl.org/dc/elements/1.1/subject> "http://umbel.org/umbel/sc/SoftwareProject"
- Sparql endpoint
 - <http://rdfs.org/ns/void#sparqlEndpoint>
- Regex pattern
 - <http://rdfs.org/ns/void#uriRegexPattern> ".*projects/sparql?query=.*" , ".*users.*"

Le tecnologie Linked Data, RDF e SPARQL vengono approfondite nel prossimo capitolo, mentre il successivo approfondisce l'architettura e le componenti della piattaforma Hackystat, i cui dati sono stati sfruttati per la creazione delle descrizioni di progetti, issue e utenti/sviluppatori, e la ricerca su di essi.

CAPITOLO II

2 LINKED DATA

I Linked Data sono la tecnologia proposta da Tim Berners Lee nel 2006 allo scopo di dirigere il Web verso un miglioramento nella gestione dei suoi contenuti. Con l'avvento del fenomeno denominato Web 2.0 e la corrispondente possibilità data agli utenti comuni di diventare creatori di contenuti in prima persona, unitamente alla digitalizzazione degli archivi cartacei che nell'ultimo decennio è aumentata costantemente, la mole di contenuti disponibili sul Web si è accresciuta enormemente. Di conseguenza il problema relativo ad una corretta gestione di tali contenuti, che riduca al minimo le risorse necessarie è diventato sempre più pressante.

La tipologia testuale di contenuti accessibili dal Web, è la più facilmente gestibile con le attuali tecnologie; mentre altri tipi di media come l'audio, il video e le immagini, presentano numerose difficoltà ulteriori e diverse tra loro, affrontabili ma non ancora in maniera del tutto efficace.

Considerato l'alto condensamento di informazione caratterizzante i contenuti testuali, nonché la più naturale rappresentazione digitale degli archivi cartacei, che costituiscono il più corposo input al processo di digitalizzazione tuttora in corso; si preferisce partire da essi per il miglioramento della gestione dell'informazione.

Attualmente la maggiore causa di inefficienza dei motori di ricerca e di dispendio risorse umane e non, risiede nella mancanza di strutturazione semantica della maggior parte dei contenuti disponibili online. Ciò accade in quanto il World Wide Web è essenzialmente costituito da “documenti” comunemente chiamate “pagine Web” e, in quanto tali, non semanticamente strutturate. La struttura imposta dall'html ad esempio, non individua univocamente e senza ambiguità sezioni del documento in base alla loro semantica. La struttura imposta tramite l'html è puramente sintattica, al più legata alla strutturazione estetica ma non semantica della pagina. La semantica che è possibile dedurre da una pagina Html si riduce alle deduzioni effettuabili in base al posizionamento strategico di una data informazione nella pagina Web. I motori di ricerca possono dunque basarsi solo sulla presenza o meno di parole chiave in un documento, che cercano di integrare con tutte le possibili informazioni deducibili dalla struttura ad Hyperlink del Web (collegamenti entranti ed uscenti da

una pagina Web rispetto ad altre pagine). Infatti l'unica alternativa consisterebbe nell'analisi del contenuto testuale in linguaggio naturale, con l'ausilio di dizionari, ma sarebbe impraticabile a causa dell'alto costo che caratterizza tale situazione e diventa ancora più insostenibile data la mole di documenti del Web.

Dalla necessità di superare il limite imposto dalla strutturazione dei contenuti online nacque il Semantic Web che non è mai decollato per i motivi che spiegheremo, partendo dalla descrizione delle varie fasi attraversate dal Web sin dalla sua creazione. I Linked Data invece, nascono da una più precisa individuazione del problema da superare, ben delineato e comprendente solo caratteristiche facilmente realizzabili, ovvero superare il Web of Document e dare l'avvio al Web of Data: il Web del futuro secondo Tim Berners Lee.

2.1 EVOLUZIONE DEL WEB

Dal momento in cui fu coniato il termine World Wide Web nel 1990 ad opera di Tim Berners Lee presso il CERN (Centro Europeo per la Ricerca Nucleare), il Web ha subito numerose trasformazioni, adattandosi alle esigenze e inclinazioni degli utenti. Fino ad oggi tali trasformazioni sono state suddivise a formare almeno due fenomeni, chiamati Web 1.0 e 2.0 mentre un terzo, chiamato Web 3.0, è in via di definizione. Il filo conduttore in tale evoluzione è la "Collaborazione" intesa come scambio utile di informazione allo scopo di accrescere la conoscenza comune. Infatti fin dal primo momento in cui due computer sono stati connessi l'un l'altro l'obiettivo è stato comunicare, scambiare informazioni per collaborare. E' significativo che Internet come lo conosciamo oggi sia nato nel 1969 e, quindi, solo dopo le e-mail, le quali invece risalgono al 1965. Il Web come descritto nel precedente capitolo, è il principale strumento di collaborazione tra sviluppatori che, in tal modo, riescono facilmente a scambiare esperienze e aumentare le possibilità e la velocità di risoluzione dei più comuni problemi incontrati durante lo sviluppo software. L'intuizione precoce di nuovi possibili miglioramenti nello sviluppo collaborativo del Software, può costituire la chiave di differenza tra un'azienda e le altre nell'universo competitivo economico internazionale.

2.1.1 PRIMA FASE: IL WEB 1.0

Prima della diffusione delle connessioni di rete che portarono alla nascita di Internet, la maggior parte di esse erano locali e il metodo di connessione preferito era il modello a mainframe. Diversi programmi di ricerca sui principi della connessione tra reti locali fisicamente separate, si evolsero fino allo sviluppo del modello di scambio pacchetti, su cui si basarono numerose soluzioni di rete

tra il 1960 e il 1970 [6] tra cui ARPANET, culla di Internet. Le esigenze di connessione di persone e conoscenze si traducono in pagine o siti “Web” contenenti informazioni su qualsiasi dominio, nella prima versione del Web negli anni '90.

Questa fase iniziale della storia del Web è stata denominata “Web 1.0” con l’unico scopo di distinguerla dalla fase successiva (si tratta di un retronimo). Il Web 1.0 si caratterizza per

- Staticità dei siti. I siti Web contenevano informazione potenzialmente utile ma che raramente subiva modifiche (in quanto solo i webmaster, utenti esperti, erano in grado di apportarle). Ciò unitamente all’assenza di interattività decrementava l’interesse dell’utente e il desiderio di replicare la visita ad un simile sito.
- Mancanza di interazione nei siti. Ai visitatori non erano forniti strumenti per contribuire alla modifica del sito, o avere una qualunque altra influenza su di esso. L’utente era quindi un “navigatore passivo” al quale era permesso, come unico ruolo attivo, l’invio di posta elettronica in formato testuale.
- Applicazioni proprietarie. I produttori di applicazioni pubblicavano software scaricabili senza consentire agli utenti di conoscerne l’evoluzione nel tempo o i dettagli di funzionamento, né di modificarli. Per esempio Netscape Navigator, seguendo la filosofia Web 1.0 era un web browser proprietario, al contrario di Mozilla Firefox che è invece Open Source.

2.1.2 SECONDA FASE: IL WEB 2.0

L’inizio della fase successiva al Web 1.0 viene fatta risalire al seguito del cosiddetto “scoppio della bolla dot-COM” (Fig. 2.1); bolla speculativa durante la quale, dal 1995 al 2001 (con un culmine il giorno 10 Marzo, 2000 con il massimo del NASDAQ a 5.132,52), i mercati azionari nei Paesi occidentali hanno visto aumentare rapidamente la crescita del nuovo settore di Internet e dei relativi campi. Il periodo è stato caratterizzato dalla fondazione (e, in molti casi, spettacolare fallimento) di un gruppo di nuove società basate sull’erogazione di servizi via Web, comunemente denominate “dot-COM”. Era sufficiente creare una qualsiasi iniziativa dot.com per ricevere facili finanziamenti in venture capital o vedere schizzare verso l’alto il valore delle azioni quotate in borsa; ma tali aziende fallirono facilmente com’erano nate. Il rinnovamento tecnologico di vasta portata è molto spesso seguito da un momento di crisi finanziaria, e le innovazioni apportate vengono assimilate soltanto successivamente.

Tuttavia non è in effetti possibile distinguere totalmente la prima fase, Web 1.0, dalla seconda, Web 2.0, in quanto esse non sono separate da una specifica innovazione tecnologica, bensì dalla nascita di nuove modalità di utilizzo e raffinamenti di tecnologie pre-esistenti. Per questo motivo per

definire il Web 2.0 si ricorre ad un insieme di applicazioni (Fig. 2.2) che condividono caratteristiche “2.0” in comune, per le quali è stato coniato il neologismo.



Fig. 2.2 – Insieme di applicazioni Web aventi caratteristiche in comune per le quali sono considerate appartenenti al Web 1.0 o al Web 2.0.

Nel Web 2.0 è l'utente stesso a creare contenuti, ad immettere nuova informazione nel Web e modificare quella già esistente. Nuove piattaforme (per esempio Myspace, Facebook, Wordpress) permettono a chiunque di creare la propria pagina personale, a costo zero, seguendo pochi, semplici e guidati passi. Ciò è reso possibile grazie ad una netta separazione tra contenuti e forma: la forma è dipendente dalla piattaforma utilizzata (Facebook ha un layout diverso da Myspace o Wordpress etc.), mentre il contenuto è inserito dall'utente. La chiave dell'incremento di diffusione della quantità di navigatori Internet e di contenuti online risiede quindi nella semplicità di utilizzo (oltre che nella diffusione delle connessioni a banda larga): la rivoluzione delle interfacce grafiche lo ha reso più vicino agli utenti.

Di conseguenza all'aumento di fruitori di Internet, di contenuti online, e di numero occorrenze di accesso ad uno stesso sito web reso più desiderabile grazie all'interattività, ogni pagina Web è diventata veicolo potenziale di pubblicità, oltre che luogo di libera espressione ad alto impatto mediatico. Allo stesso tempo la diffusione dei sistemi di social networking (Facebook, Twitter, Badoo, etc.) in cui il desiderio di instaurare amicizie con altri utenti porta ad una volontaria minore

attenzione alla salvaguardia della propria privacy [7], fornisce alle campagne pubblicitarie informazioni sui consumatori sempre più dettagliate e facilmente reperibili dai loro profili sui social network.

2.1.2.1 FILOSOFIA OPEN SOURCE

Infine nella filosofia Web 2.0 rientra la diffusione dei progetti software Open Source a cui chiunque può contribuire apportando modifiche, avanzare richieste o suggerimenti, visualizzare i dettagli di implementazione. Per esempio Netscape può essere considerato il confine standard del Web 1.0, mentre Google il confine standard del Web 2.0. Il prodotto di punta di Netscape era il browser web, un'applicazione desktop, e la strategia adottata mirava a sfruttare il dominio nel mercato dei browser per alzare il prezzo dei prodotti in vendita. Il controllo sugli standard per la visualizzazione dei contenuti e delle applicazioni nel browser, avrebbe potuto dare a Netscape un potere di mercato pari a quello raggiunto da Microsoft nel mercato dei PC. Netscape tentò anche di estendere il mercato ai web server, fornendo ai provider che avessero acquistato Netscape, la possibilità di inviare contenuti agli utenti Netscape. Infine però, sia i web browser che i web server furono soppiantati da servizi web liberamente fruibili.

Google invece, sin dalla nascita è stato un'applicazione web, pubblicata come servizio liberamente fruibile sul Web, ricevendo pagamenti solo dai clienti che desiderassero utilizzarlo privatamente. Nessuna caratteristica della filosofia delle industrie software Web 1.0 è presente. Software release programmate soppiantate dal miglioramento continuo; libero utilizzo invece che licenze o vendita; collezione di server di base, scatole nere per gli utenti/clienti, piuttosto che porting del sistema su varie piattaforme, affidando l'esecuzione in locale all'utente. Tale base non comprende solo collezioni di tool ma anche una base di dati specializzata: senza i dati i tool sono inutili, senza il software i dati non sono gestibili [8]. Le licenze software e il controllo sulle API diventano irrilevanti in quanto non ci sarà mai la necessità di distribuire il software, bensì solo di eseguirlo; inoltre senza la capacità di collezionare e gestire i dati, il software è inutile. Infatti il valore del software è proporzionale alla quantità e dinamismo dei dati di cui supporta la gestione. Mentre sia Netscape che Google possono essere definite Software Company le differenze sono talmente nette da definire due mondi diversi: nel primo rientrano Lotus, Microsoft, Oracle, SAP e altre compagnie nate durante la "rivoluzione software" del 1980; mentre il secondo comprende eBay, Amazon, Napster, DoubleClick, Akamai e altre aziende più recenti.

2.1.2.2 INTELLIGENZA COLLETTIVA

Il successo delle applicazioni Web 2.0 è fondato su ciò che Chris Anderson chiama “la lunga coda”, ovvero il potere collettivo di piccoli siti che contribuiscono ognuno a costruire dal basso il contenuto del Web. Per raggiungere il successo è necessario decentralizzare (principio alla base anche delle reti P2P).

L’opportunità offerta a chiunque di contribuire alla costruzione del contenuto Web, supporta lo sviluppo della cosiddetta “Intelligenza Collettiva” ovvero conoscenza condivisa, sfruttabile da chiunque. Alla base vi è la struttura ad hyperlink del Web. Nuovi contenuti o siti aggiunti dagli utenti vengono integrati subito nella struttura tramite altri utenti che li scoprono e creano link ad essi. E’ possibile paragonare il processo a quello delle sinapsi del cervello umano: le connessioni tra sinapsi diventano più forti a seguito della loro ripetizione e intensità; allo stesso modo le connessioni web crescono organicamente come conseguenza dell’attività collettiva svolta da tutti gli utenti web. L’intelligenza collettiva riguardante sviluppatori, progetti SW ed issue fornisce grande sostegno e nuove possibilità di evoluzione allo sviluppo software collaborativo, in cui ognuno può avvalersene e allo stesso tempo condividere.

2.1.2.3 BASI DI DATI PROPRIETARIE

Ogni significativa applicazione finora è sempre stata fondata su una base di dati specializzata: il web crawl di Google, la directory Yahoo!, la base di dati per prodotti di Amazon, la base di dati per prodotti e venditori di eBay, etc. La gestione delle basi di dati è una competenza chiave delle organizzazioni Web 2.0, al punto che spesso il termine “software” viene sostituito con “infoware”. Ma chi controlla tali dati? Nella storia di Internet si sono verificate numerose correlazioni tra controllo dei dati e dominio sul mercato (per esempio il monopolio governativo sulla registrazione dei nomi a dominio). Mentre, come precedentemente precisato, il controllo sulle API è diventato più arduo nell’era di Internet, il controllo dei dati chiave non lo è, soprattutto se tali sorgenti dati sono costose da creare oppure è difficile trarre profitto dall’effetto rete. Tuttavia esempi recenti sono significativi nel processo di verifica vantaggi e svantaggi di un completo controllo sui dati di base. Per esempio consideriamo MapQuest e Google Maps.

Su ogni mappa fornita da MapQuest vi è l’indicazione “Maps copyright NavTeq, TeleAtlas” contenente nomi di compagnie che hanno investito molto sui loro database di indirizzi stradali e direzioni. Ma rendere proprietari i propri dati si è dimostrato causa di troncamento della competitività di un prodotto. Al contrario permettere agli utenti di contribuire all’estensione dei dati, annotando manualmente le mappe o permettendo loro di crearne di nuove, rende più difficile

l'ingresso di competitori nel mercato. Google Maps ha abbracciato tale filosofia, e attualmente predomina. La diffusione è avvenuta anche grazie alle API che permettono l'integrazione dei servizi Google Maps in altre pagine, mescolandosi ad altri contenuti. Tuttavia esistono esempi opposti ed è quindi difficile dedurre una regola generale. Il controllo sui dati ha delle naturali ripercussioni sulla privacy, e frequenti sono i tentativi di infrangerla o renderla più flessibile.

Ulteriore dimostrazione dell'importanza dei dati, della loro libera fruizione, e connessione, un altro fenomeno caratteristico del Web 2.0 è la diffusione dei mashup: siti o applicazioni web di tipo ibrido, cioè tali da includere dinamicamente informazioni o contenuti provenienti da più fonti. Un esempio potrebbe essere un programma che, acquisendo da un sito web una lista di appartamenti, ne mostra l'ubicazione utilizzando il servizio Google Maps per evidenziare il luogo in cui gli stessi appartamenti sono localizzati.

2.1.2.4 DAGLI OPEN AI LINKED DATA

Così come il software proprietario ha portato al movimento per il Free Software, allo stesso modo oggi le basi di dati proprietarie stanno portando al movimento per i Free Data. I primi segnali della diffusione di progetti basati su Open Data sono emersi con la popolarità di Wikipedia, Creative Commons, e progetti software come Greasemonkey, che permette agli utenti di controllare come i dati vengono visualizzati sul proprio computer.

L'evoluzione dagli Open Data ai Linked Data è tuttora in fase di svolgimento, nonostante si sia già diffuso un termine, Web 3.0, per denominare questa nuova fase. Prima di approfondire i Linked Data nel prossimo paragrafo si descriverà il Web Semantico (sfondo fallimentare al passaggio da Web 2.0 a Web 3.0) allo scopo di evidenziare errori da non ripetere, limiti da superare e tecnologie da recuperare. Ciò è utile alla comprensione delle idee che hanno portato la nascita dei Linked Data. Infatti sia essi che il Semantic Web sono stati promulgati entrambi da Tim Berners-Lee rispettivamente nel 2006 e nel 2001: i Linked Data sono successivi e, quindi, la soluzione proposta ai problemi incontrati dal Semantic Web, che ne hanno ostacolato una piena diffusione.

2.1.3 IL WEB SEMANTICO

Il Web Semantico nonostante la sua architettura e dinamiche ben definite e pianificate, non è stato mai adottato su larga scala; a differenza dell'estesa diffusione del poco pianificato cosiddetto Web 2.0. Analizzeremo alcune motivazioni sociali e tecniche che hanno determinato il fallimento della diffusione del Web Semantico su larga scala. Ulteriori informazioni e propositi di ricerca per il

miglioramento del Web Semantico tramite l'integrazione con caratteristiche del Web 2.0 (per esempio applicando la Actor-Network Theory e risultati di ricerche sulle infrastrutture di informazione) sono inclusi in [9].

Il Web Semantico è stato considerato in passato il passo successivo nell'evoluzione del World Wide Web. Venne presentato pubblicamente nell'articolo "The Semantic Web" di Tim Berners-Lee (TBL), J. Hendler e Ora Lassila, nel 2001; mentre i primi articoli riguardanti un web comprensibile dai calcolatori in cui vengono utilizzate tecnologie per la condivisione e il riutilizzo dei dati furono presentati da TBL durante la "prima" conferenza sul WWW nel 1994, ed erano rivolti principalmente ad un pubblico scientifico di esperti. Da allora sono stati soprattutto gli esperti del settore ad essersi occupati dello sviluppo di protocolli e standard necessari a indirizzare i servizi e il web verso una nuova versione orientata alla semantica; senza essere mai realmente interessati a raggiungere gli utenti comuni. Nonostante il successo nello sviluppo dei protocolli necessari all'implementazione del web semantico, il loro grado di diffusione e utilizzo da parte di altre applicazioni e utenti è stato sempre molto basso.

La strategia del Web Semantico mira a creare un mezzo di comunicazione universale per lo scambio dei dati; interconnettendo gestione di informazioni personali, integrazione di applicazioni professionali e la condivisione globale di contenuti culturali, scientifici e commerciali [11]. Lo sviluppo delle tecnologie del Web Semantico (Fig. 2.3) può considerarsi un progetto guidato dagli standard e basato su una metodologia top-down. Tale sviluppo ha impegnato la maggior parte dell'attività di ricerca nei gruppi di lavoro del World Wide Web Consortium (W3C). Il W3C è infatti strutturato in gruppi di lavoro, interessi e coordinamento, comprendenti membri dell'organizzazione, staff a tempo pieno ed esperti del campo esterni, invitati. La sua missione è "Indirizzare il World Wide Web verso il dispiego della totalità delle sue potenzialità, tramite lo sviluppo di protocolli e linee guida che assicurino una crescita del Web a lungo termine." Tra i gruppi di lavoro del W3C vi sono in particolare:

- Semantic Web Health Care and Life Sciences Interest Group (HCLS IG), la cui missione è principalmente lo sviluppo e il supporto all'utilizzo delle tecnologie del Web Semantico nel dominio di applicazione della scienza medico sanitaria, in particolare delle scienze biologiche in quanto tali aree possono maggiormente trarre vantaggio dall'interoperabilità tra sorgenti di dati eterogenee, in coordinamento con il supporto alle decisioni. La nascita risale al 2005 e se ne stima la chiusura nel 2011.
- Semantic Web Interest Group (SWIG). Forum per lo scambio e la discussione di idee ed applicazioni innovative per il Semantic Web. Fornisce inoltre supporto agli sviluppatori

nell'utilizzo dei protocolli del web semantico e tecnologie quali RDF, OWL, SPARQL, etc. Costituisce la continuazione del lavoro svolto dall'RDF Interest Group, cominciato nel 2001, e se ne stima la chiusura per il 2011.

2.1.3.1 TECNOLOGIE PRINCIPALI

Nella visione del W3C, il Web Semantico potrà essere realizzato solamente attraverso una stratificazione di più livelli, ciascuno dei quali sarà caratterizzato da un proprio linguaggio il quale avrà il compito di estendere e completare i servizi offerti dallo strato subito inferiore, presentando al livello superiore nuove funzionalità (Fig. 2.3). Il Web è stato originariamente concepito per essere fruibile dall'uomo e sebbene tutti i suoi contenuti sono leggibili dal calcolatore, essi non sono però comprensibili dal calcolatore. L'analisi del contenuto testuale è un processo lungo e dispendioso in termini di consumo delle risorse di tempo e spazio. Le grandi proporzioni del Web rendono difficoltosa la gestione di tali contenuti sia manuale che automatizzata. La soluzione proposta consiste nell'utilizzo di metadata per descrivere le informazioni sul Web.

I metadata sono alla base di tutto il Web Semantico. I metadata sono dei “dati sui dati”: informazioni relative ai dati, tramite le quali è possibile ricavare delle informazioni sulla risorsa a cui sono associate. Per esempio un catalogo di pubblicazioni è un metadata in quanto descrive le pubblicazioni. Nel contesto specifico metadata significa “informazioni che descrivono risorse Web”. Ad ogni risorsa disponibile sul web dovrebbe essere associata una precisa descrizione. Sono stati proposti diversi schemi di metadata; allo stato attuale uno dei più diffusi è il Dublin Core, un sistema di metadata costituito da un insieme minimale di elementi per descrivere materiale digitale accessibile via rete [13]. Il set minimo è costituito da 15 elementi: Titolo (Title); Creatore (Creator); Soggetto (Subject); Descrizione (Description); Editore (Editor); Autore di contributo subordinato (Contributor); Data (Date); Tipo (Type); Formato (Format); Identificatore (Identifier); Fonte (Source); Lingua (Language); Relazione (Relation); Copertura (Coverage); Gestione dei diritti (Rights). Resource Description Framework è la base per la manipolazione dei metadata, il cui scopo principale è la definizione di un meccanismo per la descrizione delle risorse senza richiedere particolari assunzioni su uno specifico dominio di applicazione, né su una corrispondente specifica semantica.

Analizziamo ognuna delle componenti dell'architettura del Web Semantico in Fig. 2.3.

- Unicode: sistema di codifica che assegna un numero (o meglio, una combinazione di bit) a ogni carattere in maniera indipendente dal programma, piattaforma e dalla lingua (e dal suo sistema di scrittura). Tramite Unicode è possibile rappresentare i caratteri usati in quasi tutte

le lingue vive e in alcune lingue morte, nonché simboli matematici e chimici, cartografici, l'alfabeto Braille, ideogrammi etc..

- URI: sta per Uniform Resource Identifier (Identificatori uniformi di risorse); un URI è una stringa che identifica una risorsa nel Web in maniera univoca: un documento, un'immagine, un file, un indirizzo email, ecc. (es. <http://www.websemantico.org/index.php>).

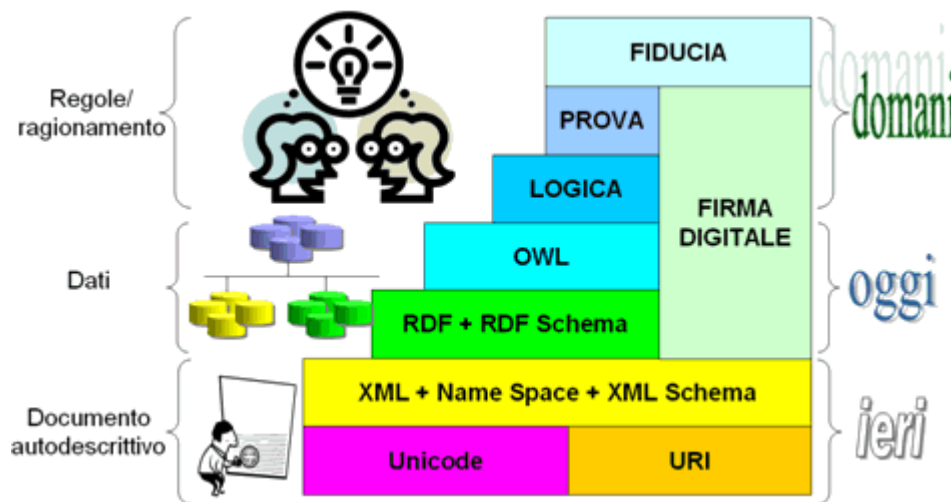


Fig. 2.3 – Modello di “torta a strati” del Web Semantico che mostra le relazioni tra le principali tecnologie del Web Semantico. [10]

- XML, Name Space e XML Schema: XML (eXstensible Markup Language) è un meta-linguaggio di markup. In pratica fornisce un insieme standard di regole sintattiche per modellare la struttura di documenti e dati. Questo insieme di regole, dette più propriamente specifiche, definiscono le modalità secondo cui è possibile crearsi un proprio linguaggio di markup. XML Schema fornisce un metodo per comporre vocabolari XML definendo le regole relative a struttura e contenuto di un documento XML. Un Namespace non è altro che un insieme di nomi di elementi e/o attributi identificati in modo univoco da un identificatore. La presenza di un identificatore univoco individua così un insieme di nomi distinguendoli da eventuali omonimie presenti in altri namespaces.
- RDF e RDF Schema: RDF (Resource Description Framework) fornisce un insieme di regole per definire informazioni descrittive sui dati, più precisamente sugli elementi costitutivi un documento web; queste asserzioni sono realizzate tramite triple (costituite da soggetto, predicato e oggetto) che legano tra loro gli elementi in una relazione binaria. RDF Schema fornisce, a sua volta, un metodo per combinare queste descrizioni in un singolo vocabolario. Il modo per sviluppare vocabolari specifici per un dato dominio di conoscenza è rappresentato dalle ontologie.

- OWL: sta per Web Ontology Language; linguaggio di markup per rappresentare esplicitamente significato e semantica di termini con vocabolari e relazioni tra i termini. Tale rappresentazione dei termini e delle relative relazioni costituisce un'ontologia. L'obiettivo è permettere ad applicazioni software di elaborare il contenuto delle informazioni dei documenti scritti in OWL. Le ontologie sono considerate tra i pilastri del Semantic Web, sebbene non abbiano una definizione univocamente condivisa (tra le definizioni possibili vi è “specifica formale di una concettualizzazione condivisa [12]). Un vocabolario del Web Semantico è un particolare tipo di ontologia detta “lightweight” o semplicemente una collezione di URI dal significato esplicitamente descritto, anche se spesso solo informalmente. Si tratta cioè di vocabolari i cui concetti e relazioni contenuti non rientrano in una gerarchia particolarmente complessa con alto numero di interconnessioni tra loro.

Fino a questo livello abbiamo brevemente identificato le tecnologie sottostanti il processo di rappresentazione della conoscenza. I gradini più elevati della piramide sono occupati da tecnologie ancora in evoluzione.

- Logica, Prova e Fiducia: Affinché il Web Semantico possa effettivamente aiutarci in una vasta gamma di situazioni, estraendo autonomamente informazioni utili dalla vasta mole di documenti web annotati semanticamente, sarà indispensabile costruire un potente linguaggio logico per realizzare le inferenze (ovvero procedimenti deduttivo mediante cui, a partire da una o più premesse, si ricava, per via logica, una conclusione). Le conclusioni ottenute saranno validate a questo livello tramite motori di validazione costituiti da sequenze di formule derivate da assiomi. Infine il sistema restituirà solo quelle informazioni che secondo il richiedente proverranno da utenti di indubbia attendibilità.

Gli altri elementi fondamentali sono rappresentati da:

- Agenti intelligenti: programmi capaci di eseguire compiti definiti da un utente in modo autonomo, ovvero senza il controllo diretto dell'utente stesso: essi raccolgono, filtrano ed elaborano le informazioni che trovano sul web;
- Firma digitale: garantisce, basandosi su di un sistema crittografico, l'autenticità delle varie asserzioni e permette di scoprire la loro provenienza. Spetta poi all'utente istruire il software del proprio computer di quali firme digitali fidarsi. Essa può essere apposta come allegato dei documenti web. L'obiettivo finale è quello che viene comunemente definito “Web of

Trust” (un web capace di offrire riservatezza, che ispiri gradualmente fiducia, e che faccia in modo che ci si prenda la responsabilità di ciò che viene pubblicato);

Riassumendo la progettazione del Web Semantico prevede che alla base vi sia una diversa e più attenta filosofia di progettazione delle risorse web, basate su XML, le quali devono rispettare gli standard definiti e recare con se una descrizione delle proprie caratteristiche tramite RDF. Ciascuna di queste risorse è identificabile in modo non ambiguo grazie all'uso degli URI (risolvendo così i problemi di ambiguità visti quando abbiamo parlato dei motori di ricerca). I metadati sono la base informativa su cui potranno operare gli agenti intelligenti per compiere le proprie azioni e prendere le proprie decisioni. Gli agenti, a loro volta, è previsto agiscano nello spazio-web sfruttando il sistema di rappresentazione della conoscenza disponibile (ontologie). Le decisioni degli agenti a questo punto saranno consentite grazie all'utilizzo di linguaggi di inferenza logica. Gli agenti, infine, nel prendere le proprie decisioni terranno conto del grado di fiducia attribuito alle risorse (ed ai loro autori identificati da sistemi di firma digitale) dagli utenti stessi.

Tali obiettivi inclusi nella progettazione del Web Semantico per la maggior parte o non hanno visto la luce (in Fig. 2.3 gli strati rientranti nella sezione “domani”) oppure non hanno avuto una diffusione su larga scala. Approfondiamo le possibili motivazioni di tale fallimento.

2.1.3.2 OSTACOLI POSTI ALLA DIFFUSIONE

Innanzitutto il problema principale riguarda le modalità di rappresentazione della conoscenza scelte nella progettazione del Web Semantico. Tale problema viene affrontato attraverso un architettura a più livelli rappresentati da:

- dati: gli elementi primitivi di informazione;
- metadati: elementi descrittivi dei dati;
- ontologie: rappresentazione semantica di dati e metadati tramite specifici linguaggi.

Questi livelli sono affiancati da una struttura tecnologica i cui principali elementi sono costituiti dall'XML, RDF, RDF Schema e OWL. Vedremo di seguito più nel dettaglio tale struttura.

2.1.3.2.1 RAPPRESENTAZIONE DELLA CONOSCENZA

L'XML ci consente di modellare la struttura di documenti e dati seguendo un insieme standard di regole sintattiche (dette più propriamente specifiche). Esso reca tra i suoi vantaggi fondamentali quello di garantire un'alta interoperabilità dei dati (e dunque di consentire l'interscambio di dati tra

piattaforme ed applicativi diversi). La struttura e la grammatica soggiacenti al documento XML possono essere determinate attraverso una DTD (Document Type Definition) o attraverso XML Schema. XML Schema fornisce un metodo per comporre vocabolari XML definendo le regole relative a struttura e contenuto di un documento XML.

Le informazioni sui dati vengono gestite principalmente attraverso RDF (Resource Description Framework), un framework, basato su sintassi XML, specificatamente proposto dal W3C per la descrizione dei metadati relativi alle risorse (ciascuna identificata da un URI). RDF fornisce in sostanza un insieme di regole per definire informazioni descrittive sui dati, più precisamente sugli elementi costitutivi un documento web. Il modello di dati RDF è rappresentato da risorse, proprietà e valori. Alla base della rappresentazione delle informazioni in RDF, come anticipato precedentemente, vi sono le “dichiarazioni” (statement) costituite da triple del tipo: Soggetto (la risorsa), Predicato (la proprietà) e Oggetto (il valore). Un modello RDF è rappresentabile da un grafo orientato sui cui nodi ci sono risorse o tipi primitivi e i cui archi rappresentano le proprietà. RDF Schema fornisce dal canto suo un metodo per combinare queste descrizioni in un singolo vocabolario. Il modo per sviluppare vocabolari specifici per un dato dominio di conoscenza è rappresentato dalle ontologie. Una volta definiti i dati e le relazioni tra questi il passaggio successivo fondamentale è rappresentato dall’attribuzione della capacità semantica a questa struttura.

Lo strumento individuato nell’ambito del web semantico per la risoluzione di questo problema è rappresentato dalle ontologie per la cui descrizione il W3C ha promosso lo sviluppo di OWL. Sebbene il concetto di ontologia venga usato in modo piuttosto generico, una vera ontologia non dovrebbe limitarsi ad una gerarchia di concetti organizzati con relazione di sussunzione ma deve prevedere anche le relazioni semantiche che descrivono le associazioni tra i concetti. Tra gli esempi più noti di ontologie disponibili possiamo citare Cyc, sistema costituito da un’ontologia costitutiva e diverse ontologie specializzate per dominio; WordNet, un progetto di rete semantica basato sui principi della psicolinguistica ed oggi adottato spesso anche come dizionario e SUMO, un tentativo di definire un’ontologia superiore avviato in seno all’IEEE.

Uno dei problemi principali di fronte a cui ci si trova davanti quando si parla di ontologie è quello della condivisione e della conciliazione di esigenze e punti di vista diversi; ovvero delle infinite visioni del mondo. Per tale motivo la generazione di un’ontologia fondante e totale risulta essere un’utopia e sempre più, anche nell’ambito del Web Semantico, si sta sviluppando un movimento di

sviluppo di ontologie provenienti dal basso, ovvero emergenti dal senso comune e dai processi sociali di negoziazione dei significati. Sempre per lo stesso motivo si tende alla creazione di diverse ontologie, ciascuna riferita ad un preciso dominio e seguente un dato punto di vista. Nasce qui l'esigenza di interoperabilità dei diversi sistemi ontologici generati, problema a cui si può ovviare perseguendo processi di standardizzazione dei linguaggi descrittivi di tali sistemi.

Nell'ambito del Web Semantico, il W3C ha sostenuto lo sviluppo di OWL (Web Ontology Language) quale linguaggio per la definizione di ontologie strutturate basate sul Web. OWL è composto da tre sottolinguaggi caratterizzati da una crescente espressività:

- OWL Lite: utile per quanti necessitano soprattutto di una gerarchia di classificazione e semplici restrizioni;
- OWL DL (Description Logics): utile per quanti ricercano il massimo dell'espressività mantenendo la completezza computazionale (tutte le conclusioni hanno la garanzia di essere calcolabili) e la decidibilità (tutte le computazioni finiscono in un tempo definito);
- OWL Full: destinato agli utenti che vogliono la massima espressività e libertà sintattica di RDF senza le garanzie computazionali.

Come indicato nei documenti ufficiali W3C "OWL Full può essere considerato come una estensione di RDF, mentre OWL Lite e OWL DL possono essere considerate come una estensione di una visione limitata di RDF. Ogni documento OWL è un documento RDF, ed ogni documento RDF è un documento OWL Full, ma solo alcuni documenti RDF saranno un documento OWL Lite oppure OWL DL".

La piena realizzazione dei principi del Web Semantico è probabilmente ancora lontana da una sua realizzazione e gli ostacoli maggiori al suo sviluppo si incontrano proprio al livello ontologico dell'architettura precedentemente vista. L'onerosità della mappatura delle risorse, la piena interoperabilità tra i diversi linguaggi utilizzati per la descrizione dei dati e le relazioni tra essi, i cambiamenti, anche culturali, profondi che si richiedono soprattutto in fase di progettazione dei documenti destinati al web, richiedono uno sforzo supplementare e quell'adeguamento sociale e tecnologico che fin dagli inizi Berners Lee aveva indicato come chiave del cambiamento.

Nel prossimo paragrafo segue un approfondimento riguardo i problemi connessi alle possibili cause per cui il Web Semantico non è ancora riuscito ad evolversi, mentre il Web 2.0 è stato oggetto di diffusione e adozione di massa tra il pubblico e le comunità di sviluppatori.

2.1.3.2.2 CONFRONTO CON IL WEB 2.0

Il principale propulsore del Web 2.0 è stato l'aspetto commerciale, mentre il Web Semantico è stato finora principalmente un progetto accademico oppure esclusivo del W3C. Diversi studi in passato hanno evidenziato l'importanza del coinvolgimento degli utenti nella rete, nella modellazione delle tecnologie [20]. La distinzione tra produttori e utenti è infatti diventata sempre più sottile e il successo dello stesso Internet può essere fatto risalire all'abilità degli utenti di creare rete per soddisfare le proprie necessità; concetto che rimarca l'attuale stato del web (Web 2.0) focalizzato sugli utenti creatori in prima persona di contenuti e condivisione.

Il Web Semantico non è un artefatto isolato bensì dovrebbe essere un'evoluzione del web esistente. La strategia per questo processo evolutivo è sempre consistita nel definire la maggior parte degli standard prima, e solo successivamente invitare la comunità ad usufruirne. Invece di incalzare la base tramite lo sviluppo di un maggior numero di servizi di attrattiva per nuovi gruppi di utenti come accaduto durante la precedente evoluzione del Web, il W3C ha scelto esattamente la strategia opposta, con risultati evidenti (Fig. 2.4).

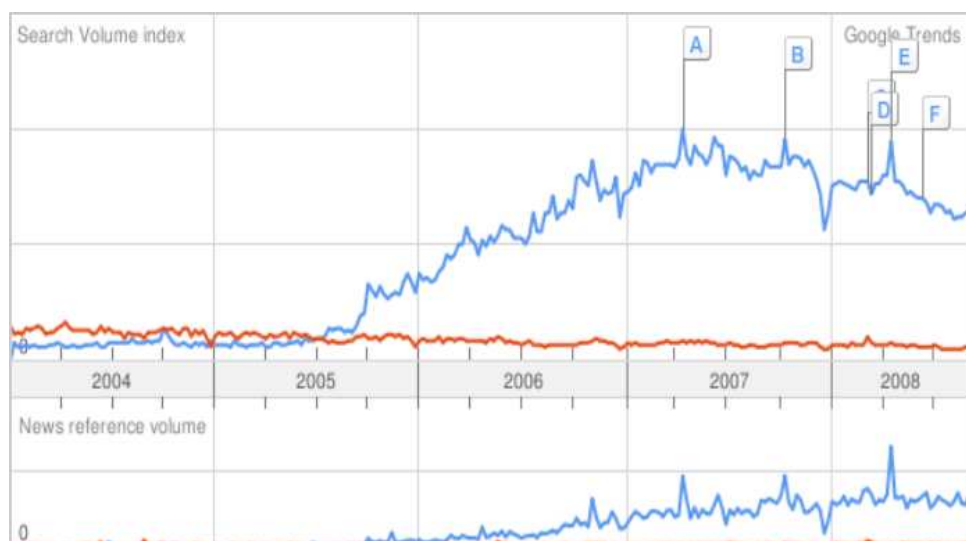


Fig. 2.4 – Tendenze rilevate nelle ricerche effettuate dagli utenti tramite Google: Web 2.0 (linea blu) vs. Web Semantico (linea rossa). L'immagine mostra come le tendenze nei volumi di ricerche e nelle quantità di riferimenti alle News si sono evolute dall'inizio del 2005 a metà 2008. [9]

Una probabile motivazione potrebbe risalire alla complessa infrastruttura organizzativa caratterizzante il W3C. Il Semantic Web è stato parte della visione di TBL del WWW sin dalle origini. Allo scopo di conservare il sistema il più possibile semplice e in grado di mobilitare quante più risorse possibile, il coinvolgimento degli utenti è sempre stato rimandato. Ciò in contrasto col metodo di design suggerito in altri studi [21], che prevede la creazione e la

coltivazione di una rete che parta dall'implementazione della soluzione più semplice e più economica possibile, la quale soddisfi le necessità degli utenti più motivati nelle loro più critiche e semplici attività e possa beneficiare il supporto alla comunicazione e collaborazione tra soli pochi utenti. Solo in seguito si consiglia l'estensione della tecnologia per coinvolgere un maggior numero di utenti inventando soluzioni più complesse e vantaggiose, passo dopo passo, ripetendo tale processo iterativamente fino al raggiungimento dell'apice della diffusione.

Non aver seguito tale strategia è una possibile causa delle difficoltà incontrate dal Web Semantico nel processo di sviluppo. La rete che si cercava di estendere infatti, aveva già oltrepassato tutte le altre infrastrutture informative e, quindi, la sua estensione avrebbe richiesto un corpo gestionale grande abbastanza per lanciare i processi di diffusione dall'interno della rete stessa; mentre il Web Semantico in quanto estensione del web esistente è solo nelle sue fasi iniziali e non raggiunge tale estensione. Ciò può spiegare il motivo per cui i poteri di gestione e standardizzazione del W3C possano aver ostacolato il processo di sviluppo del Web Semantico; ed evidenzia il dilemma con cui il W3C si scontra dovendo assolvere contemporaneamente sia ad una funzione di corpo gestionale della rete più grande del mondo, e sia ad una funzione di responsabilità verso lo sviluppo di nuovi standard necessari alla continua evoluzione delle infrastrutture informative[9]. Tale accentramento di ruoli si traduce nella classica tensione tra standardizzazione e flessibilità.

In conclusione i maggiori limiti del Web Semantico derivano da

- Difficoltà nell'utilizzo pratico delle rappresentazioni della conoscenza previste (in particolare le ontologie)
- Sviluppo di tecnologie e standard che non è mai stato mirato al coinvolgimento degli utenti. Gli utenti non sono stati fautori della costruzione del Web Semantico come accaduto nella long tail del Web 2.0.

Tim Berners Lee con la sua proposta riguardante i Linked Data nel 2006 [22], ha ridimensionato gli iniziali eccessivamente pretenziosi obiettivi del Web Semantico, focalizzando l'attenzione su quella parte di Web Semantico effettivamente implementabile e realisticamente diffondibile sul Web attuale; evidenziando i vantaggi offerti agli utenti comuni nel miglioramento della propria esperienza di navigazione sul web. Si comincia quindi ad attuare lo sviluppo graduale che coinvolga nel tempo sempre più utenti e parta da semplici cambiamenti, cui si accennava in precedenza. La parte di Semantic Web effettivamente implementabile parte dal basso in Fig. 2.3 e arriva fino ad RDF Model e Schema; al più includendo anche ontologie "lightweight". Abbandonate le ontologie

più complesse e costose, si propone l'utilizzo di vocabolari più leggeri, meno articolati, ma che continuino a svolgere la loro funzione di supporto ai metadata tramite RDF.

2.1.4 LINKED DATA

I Linked Data supportano la visione di TBL riguardo il passaggio da Web of Documents, ovvero il Web contenente documenti non strutturati semanticamente, al Web of data, ovvero il Web contenente basi di metadata distribuite e interconnesse e a loro volta associate a meta-descrizioni.

I Linked Data corrispondono, secondo una definizione del loro stesso inventore TBL, “il Web Semantico fatto bene” [30], e sono essenzialmente dati rispondenti a quattro principi:

1. Utilizzo di URI;
2. Utilizzo di URI http;
3. Fornire utili informazioni tramite SPARQL e gli standard RDF, nel momento in cui si visita un URI tramite browser;
4. Menzionare, nei metadata di una risorsa, URIs di risorse esterne relazionabili con essa.

L'obiettivo principale del Web Semantico consiste nel rendere i dati machine-understandable. A tale obiettivo già raggiunto tramite RDF, focalizzando maggiormente l'attenzione sull'interoperabilità dei dati invece che solo sui dati stessi, i Linked Data aggiungono un nuovo obiettivo: il superamento dei data silos. Nel Web 2.0 come spiegato le basi di dati sono spesso proprietarie e offrono accesso ai propri dati al più tramite XML o JSON API.

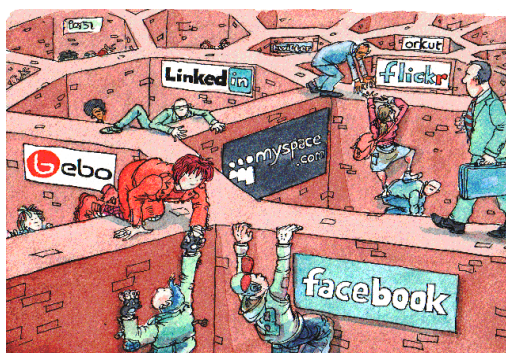


Fig. 2.5 – La maggior parte delle applicazioni Web 2.0 non offrono libero accesso ai propri dati.

L'immagine raffigura metaforicamente i dati come esseri umani e gli impedimenti al loro passaggio da un'applicazione all'altra, come muri di cinta [31].

Aggregare dati provenienti da applicazioni differenti, che espongono differenti API XML o JSON è estremamente difficoltoso. E' necessario modificare le modalità di aggregazione per adattarle al

caso specifico; inoltre la sintassi XML ostacola l'aggregazione di dati. Gli stessi mashup incontrano tali difficoltà e ne esistono numerosi a causa del loro legame col dominio specifico.

2.1.4.1 SUPERAMENTO DEI DATA SILOS

Per quale motivo le organizzazioni le cui basi di dati sono proprietarie, dovrebbero essere interessate a rendere pubblici i propri dati?

API per il pubblico accesso ai dati sono attualmente molto diffuse nel web; siti quali Amazon e Flickr costituiscono un esempio. Sia per Amazon che Flickr i dati rappresentano uno strato cardine dell'organizzazione, ma aver reso libero l'accesso tramite API ha portato loro dei vantaggi quali la nascita di comunità che utilizzano tali dati e per effetto attraggono profitti da altre aree; quali ad esempio la vendita di prodotti su Amazon o la sottoscrizione di una tassa annuale per ottenere maggiori funzionalità su Flickr. Le API hanno quindi permesso ad Amazon e Flickr di aumentare il traffico verso le loro piattaforme di profitto.

Un'organizzazione in cui i dati costituiscano una risorsa critica ha quindi due possibilità:

1. Rendere liberamente accessibili i propri dati. Comprendere che oggi la sfida non consiste semplicemente nel possedere dati di qualità ma anche nel permettere agli utenti di creare valore aggiunto su di essi e, di riflesso, sull'organizzazione.
2. Mantenere i propri dati proprietari, mentre i dati e la stessa organizzazione diventano lentamente irrilevanti, soppiantati da fonti alternative, liberamente accessibili e meno costose.

Nell'attuale scenario in cui la scelta verte tra i dati liberi o i dati obsoleti, i Linked Data svolgono un ruolo centrale[32]. Tornando all'esempio riguardante Amazon e Flickr, entrambi hanno reso i loro dati facilmente accessibili, ma impongono l'obbligo per i fruitori dei loro dati di inserire link al loro sito di origine, col conseguente aumento di traffico verso i data provider. Un tipico scenario Web 2.0 comprende dati accessibili tramite un'API, manipolati e ripresentati all'utente in una forma in qualche modo differente dall'originale fornita dal data publisher: un mashup. Le differenze possono riguardare il layout oppure valore aggiunto derivante dalla combinazione di dati provenienti da sorgenti informative diverse. In entrambi i casi questo tipo di mashup viene presentato all'utente come un documento html, eventualmente associato ad Ajax per migliorare l'interazione con l'utente. Non sempre il creatore del mashup ha interesse ad inserire effettivamente un collegamento alla sorgente di origine, ma anche nel caso in cui ciò avvenisse sia lui che il data publisher si troverebbero in una situazione svantaggiosa. Infatti la connessione tra dati e data

provider è fallace come conseguenza delle tipiche modalità di pubblicazione dati delle Web API: XML, senza neanche l'utilizzo delle URI. Per esempio sul Servizio di E-commerce Amazon il libro "Harry Potter and the Deathly Hallows" è descritto nel modo seguente:

```
<itemAttributes>  
<author>J. K. Rowling</author>  
<creator Role="Illustrator">Mary GrandPré</creator>  
<manufacturer>Arthur A. Levine Books</manufacturer>  
<productGroup>Book</productGroup>  
<title>Harry Potter and the Deathly Hallows (Book 7)</title>  
</itemAttributes>
```

Elementi quali "author" contengono il nome dell'autore come stringa testuale; l'autore non viene univocamente identificato in un modo tale che altri sorgenti dati sul Web possano farvi riferimento. L'autore specificato ha senso unicamente nel contesto di questo particolare documento che descrive un particolare libro; non vi è un URI che può essere consultato per ottenere maggiori informazioni sull'autore. Nei dati non c'è niente che porti altrove, né colleghi alla sorgente: la connessione tra dati e data publisher viene persa.

Tale connessione può essere rinforzata da link html alle sorgenti informative ma questo tipo di link sono influenzabili da incentivi economici o obblighi di licenza. Nei mashup in stile Web 2.0 basati su tali principi non esiste alcun modo affidabile per esprimere le relazioni tra le varie sorgenti di dati in modo che possano essere riutilizzate nella costruzione di ulteriori mashup: il lavoro protratto è usufruibile solo dagli uomini e solo nei limiti dell'applicazione per cui è stato creato; oltre va perso.

I Linked Data mashup invece sono semplicemente statement che collegano item provenienti da data set correlabili. Tali item sono identificati da URI http ognuno dei quali può eventualmente portare, se visitato tramite browser, nel dominio del data publisher, causando il desiderato aumento di traffico verso le proprie piattaforme di profitto.

Invece che rilasciare i dati in forma non tracciabile e anonima, i Linked Data permettono alle organizzazioni e agli individui di esporre i loro dati critici in modo che possano essere facilmente fruibili da altri utenti conservando allo stesso tempo indicatori di provenienza e strumenti per capitalizzare su o altrimenti beneficiare da, il loro contributo alla "liberazione" dei dati.

In tal modo i Linked Data creano valore anche per le imprese.

Nei prossimi paragrafi vengono approfonditi rispettivamente

1. i limiti dell'XML correlati ai limiti degli attuali mashup appena descritti, che hanno portato alla scelta di RDF come tecnologia alla base dei Linked data
2. gli Unified Resource Identifier
3. modello, sintassi e schema RDF
4. il linguaggio di query su RDF Sparql

2.1.4.1.1 XML OSTACOLO ALL'AGGREGAZIONE DATI

XML non è in grado di supportare l'interoperabilità semantica perché innanzitutto l'XML è mirato alla struttura dei documenti e non impone alcuna comune interpretazione dei dati contenuti in un documento. Poiché XML si limita a descrivere la grammatica non c'è alcun modo per riconoscere un'unità semantica proveniente da un particolare dominio di interesse.

Inoltre aggregare dati a causa di un'eventuale aggiunta di nuove entità ad una relazione pre-esistente, oppure a causa della disponibilità di nuove sorgenti informative etc. in caso si utilizzino XML e DTD o XML Schema richiede un costo molto più alto del necessario: non è sufficiente mappare un modello di dominio in un altro in quanto i modelli di dominio sono codificati nel DTD e devono prima essere re-ingegnerizzati. Una mappatura diretta basata sui diversi DTD non è praticabile perché l'obiettivo non è mappare una grammatica nell'altra, bensì mappare oggetti e relazioni da un dominio di interesse ad un altro. Quindi è necessario definire la mappatura tra differenti modelli di dominio e non tra differenti DTD. La definizione della mappatura tra DTD avviene solo successivamente. Riassumendo per aggregare dati XML provenienti da differenti sorgenti è necessario attuare la seguente procedura:

1. Re-ingegnerizzare del modello di dominio originale nel DTD o XML Schema;
2. Stabilire una mappatura tra le entità nel modello di dominio: i concetti e le relazioni provenienti dai modelli di dominio devono essere mappati l'uno nell'altro;
3. Definire procedure di traduzione per i documenti XML: poiché i documenti XML vengono scambiati, le mappature stabilite al passo precedente devono essere tradotte in procedure di mappatura per i documenti XML (per esempio definizioni XSLT per le grammatiche).

Anche questo passo implica un'elevato costo in quanto dipende dalla particolare codifica scelta per costruire i DTD iniziali. Tale elevato costo consiste nella traduzione del modello di dominio originale in un XML-DTD, la re-ingegnerizzazione del modello di dominio e la generazione delle procedure di mappatura per documenti XML basate sulle mappature di dominio stabilite.

E' evidente quindi la necessità di un formalismo più adatto dell'XML al trasferimento dati, che non renda più necessarie le traduzioni. XML è particolarmente appropriato per lo scambio di dati tra applicazioni nel caso in cui entrambe consapevoli della struttura dei dati che si stanno scambiando, ma non è adatto nel caso in cui frequentemente si verifica l'ingresso di nuovi partner di comunicazione.

Per questo motivo I Linked Data non possono basarsi solo su XML.

I limiti dell'XML vengono superati dall'utilizzo del Framework per la Descrizione delle Risorse, ovvero RDF unitamente all'utilizzo di link RDF, come riportato nei principi dei Linked Data elencati

RDF supporta l'interoperabilità semantica perché

- Le unità semantiche vengono delineate spontaneamente nella struttura oggetto-attributo che caratterizza RDF: tutti gli oggetti sono entità indipendenti;

Un modello di dominio che definisca gli oggetti e le relazioni di un dominio di interesse può essere rappresentato naturalmente in RDF, per cui i passi di traduzione necessari utilizzando XML possono essere evitati. Per trovare le mappature tra due differenti descrizioni RDF è possibile applicare direttamente tecniche di rappresentazione della conoscenza. RDF consiste di un insieme di relazioni (triple) e grazie all'utilizzo di URI è molto semplice mescolare insiemi di triple, rendendo quindi RDF ideale per l'integrazione di informazioni potenzialmente eterogenee nel Web. Inoltre i dati RDF continuano ad essere validi anche quando lo schema associato cambia; a differenza dei dati XML che sono dipendenti dal loro XML Schema.

2.1.4.2 URI: UNIFORM RESOURCE IDENTIFIER

Se vogliamo intraprendere una conversazione, o vogliamo scrivere un testo qualsiasi, dobbiamo prima identificare in maniera univoca l'argomento che vogliamo trattare, altrimenti non potremo riferirci ad esso.

Nel Web Semantico è stato definito [24] un sistema di identificatori unificato: sono gli Uniform Resource Identifiers (URI). Il nome deriva dal fatto che ogni elemento identificato viene considerato una risorsa. Gli URI sono utilizzati da RDF per codificare l'informazione in un documento, ed assicurano che i concetti non sono solo parole in un documento, ma sono vincolanti. Gli URI costituiscono la tecnologia di base ideale con la quale costruire un Web globale. Possiamo definire un URI per un qualsiasi oggetto, e qualsiasi cosa che ha un URI può essere considerato sul Web. Gli URI sono il fondamento del Web : mentre ogni parte del Web stesso può essere

rimpiazzata, gli URI no. Anche per identificare le pagine sul Web utilizziamo identificatori: sono i tipi più comuni di URI, gli indirizzi URL (Uniform Resource Locator). Guardando più in profondità si può notare che un URL comunica al computer dove trovare una risorsa specifica. Diversamente da altre forme di URI, un URL allo stesso tempo identifica e localizza. Poiché il Web è troppo esteso per essere controllato da una qualsiasi organizzazione, gli URI in massima parte sono decentralizzati. Nessuna persona o organizzazione controlla chi li produce o cosa ne fa. Questa flessibilità rende gli URI potenti, ma porta alcuni problemi. Ad esempio, poiché chiunque può creare un URI, inevitabilmente avremo più URI che rappresentano la stessa cosa; e non c'è modo per determinare se due URI puntano alla stessa risorsa. Perciò non siamo in grado di dire con esattezza cosa significa un URI.

Una pratica comune per creare URI è quella di iniziare da una pagina Web. La pagina descrive l'oggetto che deve essere identificato e spiega che l'URL della pagina è l'URI per tale oggetto. Il punto d'arrivo sarà che qualsiasi istanza rappresenterà sia la risorsa fisica, sia la pagina Web che la descrive. Ciò è noto come problema dell'identificazione delle pagine Web.

Questo è un fatto importante da comprendere. Un URI non è un insieme di direttive che indicano al computer dove trovare un file specifico nel Web (sebbene lo faccia anche), ma è un nome per una risorsa (una cosa), accessibile o meno attraverso Internet. L'URI può o no fornire un modo per ottenere più informazioni su una risorsa. Altri metodi per fornire informazioni sugli URI e le risorse che essi identificano sono in via di sviluppo. È anche vero che l'abilità di dire cose su di un URI è una parte importante del Web Semantico. Ma non dobbiamo assumere che un URI fa qualcosa di più che fornire un identificatore per una risorsa.

David Connelly del W3C ha realizzato una pagina [25] in cui ha raccolto gli schemi URI che sono stati definiti.

2.1.4.3 RESOURCE DESCRIPTION FRAMEWORK

Abbiamo già sottolineato come fosse difficile automatizzare il Web restando ancorati alla sua architettura originaria, in cui tutte le informazioni erano machine-readable, ma non machine-understandable, e come la soluzione al problema sembri venire dai metadati. L'uso efficace dei metadati, tuttavia, richiede che vengano stabilite delle convenzioni per la semantica, la sintassi e la struttura [23]. Le singole comunità interessate alla descrizione delle loro risorse specifiche definiscono la semantica dei metadati pertinenti alle loro esigenze. La sintassi, cioè l'organizzazione sistematica dei data element per l'elaborazione automatica, facilita lo scambio e l'utilizzo dei metadati tra applicazioni diverse. La struttura può essere vista come un vincolo formale

sulla sintassi, per una rappresentazione consistente della semantica. RDF (Resource Description Framework) è lo strumento base per la codifica, lo scambio e il riutilizzo di metadati strutturati, e consente l'interoperabilità tra applicazioni che si scambiano sul Web informazioni machine-understandable.

I settori nei quali RDF può essere utilizzato e portare vantaggi sono i più disparati, basti citare, a titolo di esempio:

- descrizione del contenuto di un sito Web, o di una pagina, o di una biblioteca digitale;
- implementazione di intelligent software agent, per lo scambio di conoscenza e un utilizzo migliore delle risorse Web;
- classificazione del contenuto, per applicare criteri di selezione;
- descrizione di un insieme di pagine, che rappresentano un singolo documento logico;
- stabilire i criteri di proprietà intellettuale delle singole pagine;
- esprimere criteri di privacy preference degli utenti e le privacy policies di un sito Web;
- con il meccanismo della digital signature, contribuire alla creazione del Web of Trust, per le applicazioni nel commercio elettronico, la cooperazione, etc..

Il Resource Description Framework (RDF), quindi, non descrive la semantica, ma fornisce una base comune per poterla esprimere, permettendo di definire la semantica dei tag XML. RDF è costituito da due componenti:

- RDF Model and Syntax: definisce il data model RDF e la sua codifica XML;
- RDF Schema: permette di definire specifici vocabolari per i metadati.

Tali componenti vengono approfonditi nei prossimi paragrafi.

2.1.4.3.1 MODELLO E SINTASSI

RDF fornisce un modello per descrivere le risorse che possono avere delle proprietà (o anche attributi o caratteristiche). Per RDF una risorsa è un qualsiasi oggetto che sia identificabile univocamente mediante un URI. L'RDF Data Model è quindi basato su tre tipi di oggetti:

- Risorse: una risorsa può essere rappresentata da una pagina, un gruppo di pagine, un'immagine, un server o una qualsiasi altro elemento che abbia un URI
- Proprietà: una proprietà è una specifica caratteristica o attributo di una risorsa; una proprietà può anche descrivere relazioni con altre risorse
- Asserzioni: una asserzione è costituita dall'insieme di una risorsa, una proprietà e uno specifico valore per quella proprietà e descrive le caratteristiche di una risorsa e le relazioni con altre risorse

Talvolta, è necessario far riferimento a più di una risorsa (un documento può essere composto da una serie di componenti). Per questo scopo RDF definisce tre tipi di container:

- Bag: è una lista non ordinata di risorse o costanti. Viene utilizzato per dichiarare che una proprietà ha valori multipli, senza alcun significato particolare attribuito al loro ordine (per esempio, i componenti di una commissione). Sono ammessi valori duplicati;
- Sequence: è una lista ordinata di risorse o costanti. Viene utilizzato per dichiarare che una proprietà ha valori multipli, e che il loro ordine è significativo (per esempio un insieme di nomi di cui si voglia preservare l'ordine alfabetico). Sono ammessi valori duplicati;
- Alternative: è una lista di risorse o costanti che rappresentano una alternativa per il valore (singolo) di una proprietà. Può essere utilizzato, per esempio, per fornire titoli alternativi in varie lingue.

È possibile definire proprietà sia dell'intero container che dei singoli elementi.

Non bisogna confondere i container e le proprietà multiple: una risorsa può essere soggetto in più asserzioni, sempre con lo stesso predicato (per esempio, *Calvino* è autore di "*Se una notte d'inverno un viaggiatore*", "*Le fiabe italiane*", "*Il barone rampante*",...). Si noti che è semanticamente diverso il caso in cui si ha una singola asserzione il cui oggetto è un container contenente vari esemplari. Per esempio, l'asserzione: "*La commissione composta da X, Y e Z ha adottato una decisione*", non implica che ogni membro della commissione abbia espresso lo stesso parere, come invece sarebbe nel caso di un'asserzione multipla.

In alcuni casi, può essere utile poter certificare la credibilità di una particolare asserzione, cioè formulare delle asserzioni relative ad altre asserzioni. Per esempio la risorsa:

<http://www.nomesito.it/Dorati/Tesina.html>

has Author Antonella Dorati

viene vista da RDF come un fatto. Invece, l'asserzione:

Alessio Crisologo dice che la risorsa

<http://www.nomesito.it/Dorati/Tesina.html>

has Author Antonella Dorati

non afferma un fatto relativo alla risorsa <http://www.nomesito.it/Dorati/Tesina.html> bensì un fatto relativo all'affermazione di Alessio Crisologo. Per esprimere questo fatto in RDF, è necessario modellare l'asserzione come una risorsa con quattro proprietà:

- **soggetto:** identifica la risorsa che viene descritta dall'asserzione modellata, quindi il soggetto è la risorsa relativamente alla quale era stato formulato l'asserzione originale (<http://www.nomesito.it/Dorati/Tesina.html>, nell'esempio)
- **predicato:** identifica la proprietà originale nell'asserzione modellata. Il valore del predicato è una risorsa che rappresenta la specifica proprietà nell'asserzione originale (nel nostro esempio, Author)
- **oggetto:** identifica il valore della proprietà nell'asserzione modellata. Il valore dell'oggetto è l'oggetto nell'asserzione originale (nel nostro esempio: "Antonella Dorati")
- **tipo** descrive il tipo della nuova risorsa

Una nuova risorsa con queste quattro proprietà rappresenta l'asserzione originale, e può essere utilizzata come oggetto di altre asserzioni e avere ulteriori asserzioni che lo riguardano. Nell'ambito della comunità che si interessa di Rappresentazione della Conoscenza, questo processo prende il nome di reificazione, e il modello di asserzione è detto “reified statement”.

2.1.4.3.2 SCHEMA

Il modello di RDF non permette di effettuare validazione di un valore o restrizione di un dominio di applicazione di una proprietà. Questo compito è svolto da RDF Schema. A differenza di XML Schema o di un DTD, RDF Schema non vincola la struttura del documento, ma fornisce informazioni utili all'interpretazione del documento stesso. Fornisce un meccanismo di base per un sistema di tipizzazione da utilizzare in modelli RDF. Lo schema è definito in termini di RDF stesso. RDF Schema definisce un insieme di risorse RDF da usare per descrivere caratteristiche di altre risorse e proprietà RDF.

- **rdfs:Resource** Tutto ciò che viene descritto in RDF è detto risorsa. Ogni risorsa è istanza della classe **rdfs:Resource**.
- **rdfs:Literal** Sottoclasse di **rdfs:Resource**, rappresenta un letterale, una stringa di testo.
- **rdf:Property** Rappresenta le proprietà. E' sottoclasse di **rdfs:Resource**.
- **rdfs:Class** Corrisponde al concetto di tipo e di classe della programmazione objectoriented.
- Quando viene definita una nuova classe, la risorsa che la rappresenta deve avere la proprietà **rdf:type** impostata a **rdfs:Class**.

- `rdfs:subClassOf` Specifica la relazione di ereditarietà fra classi. Questa proprietà può essere assegnata solo a istanze di `rdfs:Class`. Una classe può essere sottoclasse di una più classi (ereditarietà multipla).
- `rdfs:subPropertyOf` Istanza di `rdf:Property`, è usata per specificare che una proprietà è una specializzazione di un'altra. Ogni proprietà può essere la specializzazione di zero o più proprietà.
- `rdfs:seeAlso` Specifica una risorsa che fornisce ulteriori informazioni sul soggetto dell'asserzione.
- `rdfs:isDefinedBy` E' sottoproprietà di `rdfs:seeAlso` e indica una risorsa che definisce il soggetto di un'asserzione

I predicati più utilizzati per esprimere vincoli su altre proprietà sono i seguenti:

- `rdfs:range` (codominio) Usato come predicato di una risorsa `r`, indica le classi che saranno oggetto di un'asserzione che ha `r` come predicato.
- `rdfs:domain` (dominio) Usato come predicato di una risorsa `r`, indica le classi (soggetto) a cui può essere applicata `r`.

Nell'esempio in Fig. 2.5 `Persona` è di tipo classe, ed è sottoclasse di `rdfs:resource`;

`http://people.com/id/1375` è di tipo `Persona`. Il suo nome ed il suo Email sono di tipo `Literal` ovvero stringa. Da notare che tutto è sottoclasse di risorsa.

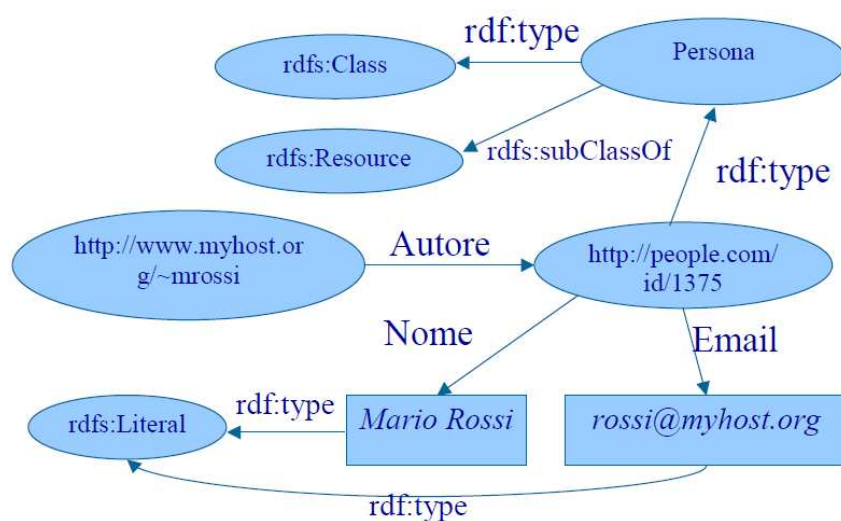


Fig. 2.5 – Esempio di rappresentazione a grafo di un RDF Schema [26].

2.1.4.4 SPARQL: LINGUAGGIO DI QUERY SU RDF

Sparql è un linguaggio di query specifico per RDF. Sull'opportunità della nascita di un tale linguaggio specifico sono state svolte delle indagini analizzando affinità e divergenze tra SPARQL, XQuery e SQL [28]. È innanzitutto possibile rilevare notevoli somiglianze tra i Data Model su cui si basano i tre linguaggi. L'RDF Data Model, ad esempio, trova corrispondenza con il modello Entity-Relationship. Per esempio l'asserzione RDF: soggetto: id1234; predicato: anno; oggetto: 1994 è facilmente "traducibile" come segmento di riga della tabella di un database relazionale nella forma "chiave - nome colonna - valore colonna". Discorso analogo vale per l'XPath Data Model, su cui è basato il linguaggio di interrogazione dell'XML, XQuery.

Se è possibile trasporre un Data Model in un altro, è altresì possibile trasporre l'uno nell'altro i rispettivi linguaggi di interrogazione. Avvalendosi di questi isomorfismi, l'analisi in [28] conclude affermando che gli scopi di SQL e di SPARQL sono abbastanza diversi tra loro da giustificare la creazione di un linguaggio specifico per l'interrogazione dell'RDF. E' tuttavia confortante la convinzione che sia possibile tradurre espressioni SPARQL in espressioni SQL, permettendo così a chi ne fa uso, di immagazzinare i propri dati RDF in database relazionali e di scrivere le query, a seconda dei casi, in SQL oppure in SPARQL; mentre sebbene sia parimenti possibile trasformare espressioni SPARQL in espressioni XQuery, difficilmente molti intraprenderanno questa strada. SPARQL adotta la sintassi Turtle, un'estensione di N-Triples, alternativa estremamente sintetica e intuitiva al tradizionale RDF/XML. Si considerino le seguenti triple RDF:

```
@prefix cd: <http://example.org/cd/>
@prefix: <http://example.org/eseempio/>
:Permutation cd:autore "Amon Tobin".
:Bricolage cd:autore "Amon Tobin".
:Amber cd:autore "Autechre".
:Amber cd:anno 1994.
```

Le asserzioni sono espresse in concise sequenze soggetto – predicato - oggetto e delimitate da un punto fermo. @prefix introduce prefissi e namespace; i due punti senza prefisso (seconda riga) definiscono il namespace di default. Gli URI sono inclusi tra parentesi angolari. I letterali di tipo stringa sono contrassegnati da virgolette.

2.1.4.4.1 PATTERN MATCHING

Le query SPARQL si basano sul meccanismo del "pattern matching" e in particolare su un costrutto, il "triple pattern", che ricalca la configurazione a triple delle asserzioni RDF fornendo un modello flessibile per la ricerca di corrispondenze [30]. Per esempio: *?titolo cd:autore ?autore* ; in luogo del soggetto e dell'oggetto questo "triple pattern" prevede due variabili, contrassegnate con ?. Le variabili fungono in un certo senso da incognite dell'interrogazione, cd:autore funge invece da costante: le triple RDF che trovano riscontro nel modello assoceranno i propri termini alle variabili corrispondenti. Un esempio di semplice query di selezione SPARQL, da cui è evidente l'analogia con SQL, è il seguente:

```
PREFIX cd: <http://example.org/cd/>  
SELECT ?titolo ?autore ?anno  
FROM <http://cd.com/listacd.ttl>  
WHERE {?titolo cd:autore ?autore.  
           ?titolo cd:anno ?ann .  
}
```

PREFIX dichiara prefissi e namespace; SELECT definisce le variabili di ricerca da prendere in considerazione nel risultato (nell'esempio: titolo, autore e anno); FROM specifica il set di dati su cui dovrà operare la query (si suppone che le triple siano immagazzinate presso l'indirizzo fittizio "http://cd.com/listacd.ttl"). È inoltre possibile ricorrere a clausole FROM NAMED e alla parola chiave GRAPH per specificare più set di dati. La clausola WHERE , infine, definisce il criterio di selezione specificando tra parentesi graffe uno o più "triple patterns" separati da punto fermo.

Applicando la query al set di triple del paragrafo precedente, si ottiene il seguente risultato:
titolo: "Amber" ; autore: "Autechre"; anno: 1994.

Il "binding" tra variabili e termini reperiti corrispondenti (in questo caso, un termine per ciascuna variabile) è reso in forma di tabella come un rapporto campo-valore: le righe rappresentano i singoli risultati, le intestazioni di cella rappresentano le variabili definite nella clausola SELECT , le celle i termini associati alle variabili.

La query precedente ha catturato esclusivamente le triple dotate di tutti e tre i termini richiesti (titolo, autore, anno), escludendo le triple che ne possedevano due soltanto (titolo, autore). È possibile riformulare la query in modo più elastico, prevedendo l'eventuale assenza di alcuni termini. Per esempio:

```

PREFIX cd: <http://example.org/cd/>
SELECT ?titolo ?autore ?anno
FROM <http://cd.com/listacd.ttl>
WHERE {?titolo cd:autore ?autore.
        OPTIONAL {?titolo cd:anno ?anno}}
}

```

Nell'esempio, il secondo pattern è dichiarato opzionale: l'informazione è aggiunta al risultato solo se disponibile, altrimenti le variabili compariranno prive di valore (unbound). Le risorse sprovviste della proprietà anno sono mostrate ugualmente e le celle dei valori mancanti sono lasciate vuote. Un altro modo per assicurare una certa elasticità nel reperimento dei dati è l'utilizzo della parola chiave UNION, che esprime un OR logico: la query non si limita pertanto alle triple che soddisfano entrambi i triple patterns, ma cattura sia quelle che soddisfano il primo, sia quelle che soddisfano il secondo.

È anche possibile porre restrizioni sui valori da associare alle variabili. Ad esempio:

```

PREFIX cd: <http://example.org/cd/>
SELECT ?titolo ?anno
FROM <http://cd.com/listacd.ttl>
WHERE {?titolo cd:anno ?anno.
        FILTER (?anno > 2000)}.
}

```

In questo caso, la restrizione è effettuata mediante l'operatore di confronto > : il filtro esclude i termini che non soddisfano la condizione definita tra le parentesi tonde: il risultato in questo caso è nullo (il dataset di riferimento non prevede valori maggiori di 2000 per la proprietà anno).

Infine come in SQL, è possibile escludere dal risultato i valori duplicati mediante la parola chiave DISTINCT, oppure ordinarli mediante la parola chiave ORDER BY, oppure limitarne la quantità mediante LIMIT.

2.1.4.4.2 FORMATO DELL'OUTPUT

Il W3C raccomanda una ben definita sintassi XML da utilizzare come formato dell'output restituito dalle query SPARQL [29]. La sintassi è intuitiva, leggibile e facile da gestire mediante fogli di stile.

Sparql è l'elemento radice, mentre l'attributo xmlns definisce, come di consueto, il namespace di riferimento. Nella sottosezione head sono elencate, nello stesso ordine in cui compaiono nella SELECT della query, le variabili da prendere in considerazione nel risultato, indicate come valori dell'attributo name degli elementi vuoti variable. La seconda sottosezione, results , contiene una sequenza di elementi result che esprimono, per ciascun risultato della query, il binding tra variabili, indicate come valore dell'attributo name dell'elemento binding, e rispettivi valori (nel caso dell'esempio, letterali di tipo stringa).

I valori booleani degli attributi ordered e distinct dell'elemento results indicano se prendere in considerazione o meno gli eventuali costrutti ORDER BY o SELECT DISTINCT della query.

Nel prossimi capitoli si approfondirà la struttura della sorgente di dati da trasformare in Linked Data, Hackystat, e successivamente si descriverà la struttura del sistema sviluppato in questa tesi, Hackystat Linked Sensor Data (LiSeD). Nell'ultimo capitolo sono contenuti i dettagli implementativi di LiSeD.

CAPITOLO III

3 IL SISTEMA HACKYSTAT

Hackystat è un framework per il rilevamento, l'analisi, la visualizzazione, l'interpretazione, l'annotazione e la disseminazione del processo di sviluppo software e dei product data [1]. Si tratta di un obiettivo ambizioso e per raggiungerlo Hackystat è organizzato come una collezione di servizi debolmente accoppiati, che comunicano utilizzando i principi architetturali REST. Un diagramma contenente un sottoinsieme dei servizi Hackystat che mostra le modalità di comunicazione tra essi, è in Figura 3.1.

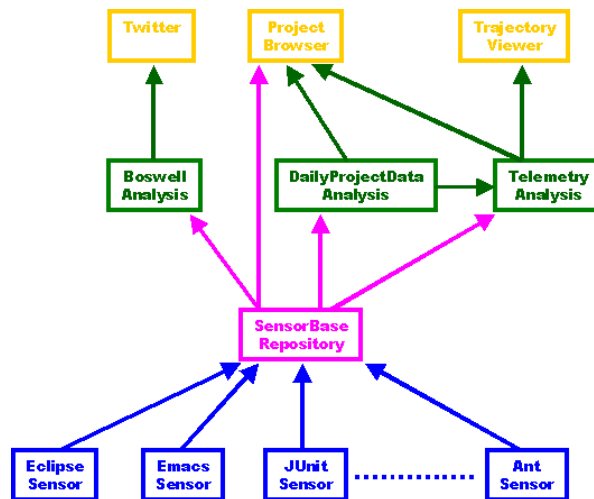


Figura 3.1 – Diagramma architetturali di Hackystat contenente solo un sottoinsieme di tutti i servizi disponibili [33].

I riquadri blu posizionati alla base dell'architettura, rappresentano i sensori. I sensori sono dei plugin che collezionano dati relativi all'utilizzo di tool di sviluppo software. Un sensore Hackystat è in grado di ricevere dati specificati direttamente dallo sviluppatore ma, prevedendo il costo che ciò richiederebbe, si evita di richiedere la specifica manuale dei dati di basso livello.

I sensori inviano i loro dati ad un repository chiamato SensorBase, (riquadro rosa in Figura 3.1) in cui vengono memorizzati. L'architettura di Hackystat supporta la compresenza di istanze multiple di SensorBase.

I sensor data tendono ad essere di basso livello e voluminosi; nella maggior parte dei casi è necessario analizzarli e astrarre su di essi per renderli utili agli sviluppatori. Per questo motivo il sistema Hackystat oltre ad essere composto dai sensori e dalla SensorBase, comprende servizi di analisi che creano delle astrazioni sui dati di basso livello (riquadri verdi in Figura 3.1). L'analisi effettuata dal servizio DailyProjectData fornisce un insieme di astrazioni generate da tutti i sensor data di basso livello dato uno specifico giorno e progetto software. L'analisi effettuata dal servizio Telemetry produce informazioni riguardo la tendenza generale dello sviluppo software, basata sui sensor data di basso livello e sulle analisi del DailyProjectData, collezionati per un dato progetto, in un intervallo temporale maggiore di un unico giorno. Boswell invece, è un servizio di analisi ancora in fase sperimentale, che supporta gruppi di lavoro distribuiti e una sorta di biografia dello sviluppatore, tramite la comunicazione con Twitter e Facebook.

Infine i riquadri gialli in Figura 3.1 rappresentano i servizi di interfaccia utente, che forniscono informazioni all'utente, prelevandole dal framework, e in alcuni casi consentono all'utente di anche di inserire informazioni nel framework stesso. Per esempio il ProjectBrowser visualizza sia i dati provenienti dalla SensorBase che le analisi del DailyProjectData, e permette all'utente di creare e

modificare i dettagli sui progetti software memorizzati nella SensorBase. Le interfacce utente non sono necessariamente limitate ai web browser, e meccanismi di interfacciamento alternativi (per esempio gli ambient device) sono in fase di studio.

Coerentemente coi principi architetturali REST tutte le frecce in Figura 3.1 rappresentano la comunicazione tra servizi in cui si utilizzano le operazioni standard HTTP GET, PUT, POST e DELETE. Una delle conseguenze è che Hackystat è neutrale a tecnologie e piattaforme. Per esempio un sensore implementato usando .NET può collezionare dati di basso livello che verranno analizzati da un servizio implementato usando Java, e i risultati possono essere visualizzati attraverso web application implementate usando Ruby on Rails. Lo stile architetturali REST si basa sul concetto di “risorsa”, ovvero un’entità che può essere “identificata” tramite un URI, e che può avere una o più “rappresentazioni”. Attualmente l’unica rappresentazione supportata dal sistema Hackystat (e quindi da tutte le sue componenti) è l’XML.

Sono state effettuate ipotesi concernenti le più comuni necessità di salvaguardia della privacy, in base alle quali i dati dei sensori associati ad un utente sono accessibili solo da tale utente precedentemente autenticatosi. Solo i propri dati possono essere utilizzati per effettuare analisi di alto livello tramite DailyprojectData, Telemetry e Portfolio, a meno che non si sia membri di un particolare progetto insieme ad altri colleghi: in tal caso le analisi vengono effettuate sui dati di tutti i membri del progetto stesso, nonostante l’accesso ai dati di basso livello continui ad essere limitato ai propri. Le regole sulla privacy sono predefinite e non è possibile in alcun modo personalizzarle. L’unico utente avente accesso a tutti i dati di tutti gli utenti è l’amministratore (unico) della SensorBase.

3.1 DATI DEI SENSORI

Per utilizzare Hackystat è necessario registrarsi su un Hackystat SensorBase server, ottenendo di conseguenza un account personale. Successivamente installando i sensori per i tool di sviluppo comunemente usati, si continuano le proprie attività di sviluppo software e i sensori invieranno al server SensorBase su cui si è registrati informazioni su tali attività, in maniera trasparente. Tramite il ProjectBrowser è possibile visualizzare i sensor data collezionati in un dato giorno per un dato progetto software.

I sensori sono piccoli plugin software applicati a tool dell’ambiente di sviluppo. Tutte le istanze di dati dei sensori sono caratterizzate da sei campi obbligatori:

Campo	Esempio	Descrizione
Timestamp	2007-11-07T09:11:12.247-10:00	Istante di tempo in cui il dato è stato collezionato dal tool o dall'utente, in formato xs:dateTime (oppure XMLGregorianCalendar)
Runtime	2007-11-07T09:11:12.247-10:00	Quando istanze multiple di sensor data dovrebbero essere aggregate, il valore temporale nel campo "Runtime" è utile ad indicare quali istanze di sensor data appartengono ad una singola collezione. Per esempio considerando un tool che genera un'istanza di sensor data per file, il sensore applicato a tale tool assegna lo stesso Runtime timestamp ad ogni istanza generata durante una singola esecuzione del tool sul sistema.
SensorDataType	DevEvent	Stringa indicante il tipo di dato.
Resource	file:/C:/svn-google/sensordata.xsd	URI indicante la risorsa da cui l'istanza di sensor data è stata generata.
Owner	johnson@hawaii.edu	Account associate con il proprietario di tale sensor data. Mentre i sensori assegnano a tale campo un semplice indirizzo e-mail, la SensorBase associa l'indirizzo ad un RESTful URI come ad esempio http://dasha.ics.hawaii.edu:9876/sensorbase/users/johnson@hawaii.edu
Tool	Eclipse	Nome del tool che ha generato il dato.

Le istanze possono comprendere anche campi opzionali che si rendessero eventualmente necessari in uno specifico dominio, per contenere informazioni aggiuntive. Per esempio per un sensore applicato ad un editor potrebbe rendersi necessario rilevare informazioni riguardo al tipo di evento di editing che ne ha causato l'attivazione; oppure nel caso di un sensore applicato ad un tool di analisi della complessità del codice potrebbe essere utile rilevare i valori di complessità. Per costruire un'analisi di alto livello in maniera efficiente ed efficace, è necessario determinare le istanze di sensor data e i relativi campi opzionali più utili a seconda del contesto. Allo stesso modo

sensori per tool aventi scopi simili dovrebbero utilizzare gli stessi campi opzionali, in modo che le analisi effettuate su un tool e sull'altro possano interoperare.

Per facilitare l'individuazione dei sensor data più utili, si definiscono delle tipologie di dati dei sensori, ognuna comprendente particolari informazioni. Il tipo è specificabile nelle istanze nel campo "sensordatatype". Sono previsti dei tipi predefiniti (standard) di sensor data:

Nome	Tool di esempio	Descrizione
Build	Ant	Rappresenta il risultato di un singolo build di un sistema.
CodeIssue	Checkstyle, PMD, FindBugs	Rappresenta un singolo file e contiene la somma di tutti i tipi di errori riscontrati in quel file.
Commit	SVN	Rappresenta il risultato di un evento di commit terminato con successo su un singolo file.
Coupling	JDepend, DependencyFinder	Rappresenta un singolo file (o directory) e una misura dell'accoppiamento tra esso e altri file (o directories).
Coverage	Clover, Emma	Rappresenta un singolo file e uno o più misure sulla copertura dei casi di test sul codice di tale file.
DevEvent	Emacs, Eclipse, Vim	Rappresenta eventi comportamentali dello sviluppatore. Ad esempio l'invocazione di un compilatore o l'apertura di un file per la modifica o l'esecuzione di uno unit test, etc.
FileMetric	SCLC, JavaNCSS	Rappresenta un singolo file e uno o più misure riguardo la dimensione o complessità di tale file.
Issue	Bugzilla, Trac, Jira, Google Project Hosting	Rappresenta la creazione di un issue in un Issue Management System.
UnitTest	JUnit	Rappresenta il risultato dell'invocazione di un singolo unit test su un file sorgente.

Ad ognuno di tali tipi standard di sensor data è previsto che siano associati dei campi opzionali ben definiti, elencati di seguito.

Tipo di	Chiave	Valori di esempio	Descrizione
---------	--------	-------------------	-------------

Sensor Data			
Build	Result	'Success', 'Failure'	Stringa “Success” o “Failure” a seconda che il build sia terminato con successo o meno.
	Target	'compile'	Target di alto livello invocato come parte del build.
	Type	'continuous.integration'	Tipo di evento di build.
CodeIssue	Type_*	10	Qualsiasi proprietà avente “Type_” come prefisso, indica un tipo di code issue. Per esempio “Type_Indentation” indica un code issue di tipo “indentazione”. Il valore è il numero di occorrenze nel file. Se non è definita alcuna di tali proprietà, allora il file non contiene code issue.
Commit	linesDeleted	10	Quantità di linee cancellate in un commit.
	linesAdded	10	Quantità di linee aggiunte in un commit.
Coupling	Type	'class', 'package'	Tipo di accoppiamento rilevato.
	Afferent	10	(Expected) The number of afferent couplings (number of program units that depend on this one.)
	Efferent	20	(Expected) The number of efferent couplings (number of program units that this one depends upon).
Coverage	*_Covered	10	(Expected) The number of covered constructs of the given coverage type.

			For example, 'method_Covered' indicates the number of covered methods.
	*_Uncovered	10	(Expected) The number of uncovered constructs of the given coverage type. For example, 'method_Uncovered' indicates the number of uncovered methods.
DevEvent	Type	'StateChange'	(Expected) The type of dev event.
FileMetric	TotalLines	200	(Expected) The total number of lines in this file.
	*ComplexityList	'12, 13, 23'	Tools such as JavaNCSS can calculate measures of complexity as well as size. This data is stored as a comma-separated list of values. The property name indicates the type of complexity. For example, JavaNCSS provides the 'CyclomaticComplexityList' property.
Issue	IssueId	'19'	(Expected) The unique ID assigned by the issue management system for this issue.
	Type	'Defect--2008-09-07T11:00:00'	The types of the issue with the time when the type updated from previous value to this value. Values of Type include Defect, Enhancement, Task, etc. D Can be multiple properties with this key.
	Status	'Open--2008-09-07T11:00:00'	The status of the issue with the time when the type updated from previous value to this value. Values of Status

			include New, Accepted, Start, Fixed, Wontfix, etc. Can be multiple properties with this key.
	Priority	'Critical--2008-09-07T11:00:00'	The priority of the issue with the time when the type updated from previous value to this value. Values of Priority include Low, Medium, High and Critical. Can be multiple properties with this key.
	Milestone	'8.4--2008-09-07T11:00:00'	The milestone of the issue with the time when the type updated from previous value to this value. Values of milestone can be any string that represent the milestone version, like 8.3. Can be multiple properties with this key.
	Owner	'philip@hawaii.edu--2008-09-07T11:00:00'	The owner of the issue with the time when the type updated from previous value to this value. Values of the owner are the hackystat account of the issue owner, mapped from account of issue tracking system with the SensorShellMap. Can be multiple properties with this key.
UnitTest	Name	'org.hackystat.TestFoo'	(Expected) The name of the unit test.
	Result	'Pass', 'Fail'	(Expected) Whether the unit test passed or failed.

Tuttavia la gestione dei tipi di sensor data è molto flessibile: non sono effettuati controlli sulla correttezza di un'istanza in base alla definizione del tipo cui è associata, ed è possibile far riferimento ad un nuovo tipo senza che esso sia stato prima definito. Creare ed utilizzare nuovi tipi di dati dei sensori è dunque estremamente semplice.

3.2 ASTRAZIONI DI TIPO PROGETTO

La SensorBase implementa un'astrazione chiamata "Progetto", che fornisce una vista particolare sui dati memorizzati, orientata ai dati riguardanti i Progetti. Lo scopo principale è supportare il lavoro di gruppo, spinti dalle seguenti motivazioni:

1. La maggior parte degli utenti lavora su più progetti contemporaneamente. Di conseguenza è utile analizzare il lavoro effettuato su ogni progetto separatamente, ed eventualmente poterne fare un confronto.
2. La maggior parte degli utenti lavora su progetti con altre persone. Di conseguenza è utile poter aggregare dati collezionati dalla propria personale attività, con quelli collezionati dai propri colleghi. Per esempio sarebbe possibile effettuare un unico build giornaliero senza dover obbligare ogni membro del team di lavoro ad effettuare il proprio build separatamente.

I progetti sono associati ai dati dei sensori in modo logico e non fisico. Di conseguenza un tool non genera sensor data esplicitamente associati ad un progetto; mentre la modifica, creazione o cancellazione dei progetti ha come unico effetto la modifica dei permessi su porzioni di sensor data che però non vengono in alcun modo modificati né cancellati.

I Progetti Hackystat sono caratterizzati da date di inizio e fine per permettere l'analisi dei dati associati, ad esempio, ad un singolo incremento di sviluppo.

Ogni utente è associato ad un progetto definito automaticamente al momento della registrazione sulla SensorBase, chiamato "Default", che raccoglie i dati collezionati da tutti i propri sensori. Esso nasce per supportare il caso in cui un utente non lavori in team con altri colleghi, oppure non abbia la necessità di analizzare il proprio lavoro separatamente a seconda del progetto cui è riferito. Nel caso in cui invece, si desideri analizzare solo porzioni dei propri dati, divisi per progetto su cui si sta lavorando, è necessario utilizzare progetti Hackystat diversi da quello di Default. E' sufficiente creare un nuovo progetto e specificarne i dettagli tra cui, in particolare, gli UriPattern. Un UriPattern è il percorso all'interno del File System in cui si trovano file correlati al progetto di interesse. In tal modo i sensori attivati in seguito ad attività effettuate su file aventi un certo percorso nel file system, inviano dati alla SensorBase che si assoceranno al progetto di interesse nel caso in cui tale percorso corrisponda ad un UriPattern per quel progetto. Altri dettagli importanti specificabili riguardo i progetti sono:

- Il proprietario (che è unico)
- Zero o più membri o spettatori.
- Data di inizio e fine del progetto.

Infatti una volta creato il progetto è possibile invitare altri membri del team o spettatori.

Confermando l'appartenenza ad un progetto, si consente implicitamente a tutti gli altri membri dello stesso di accedere ai propri dati personali. A causa delle conseguenti implicazioni sulla privacy Hakystat adotta i seguenti criteri riguardo l'appartenenza ad un progetto:

- L'utente che crea il progetto ne è considerato il "proprietario", il solo che possa anche cancellarlo, rimuoverne i membri, aggiungere e rimuovere spettatori. Egli tuttavia può solo invitare altri utenti come membri, e non aggiungerli direttamente.
- Un utente che riceva un invito a partecipare ad un progetto può accettarlo diventandone membro oppure declinare.
- Ogni membro di un progetto può effettuare analisi sui dati associati a quel progetto e collezionati da tutti i suoi membri. In qualsiasi momento è possibile cancellarsi dalla lista membri.
- Il progetto "Default" non è modificabile
- Uno spettatore è un utente che può solo effettuare analisi sui dati del progetto ma i cui dati non vengono aggregati a quelli degli altri membri durante le analisi. In qualsiasi momento può cancellarsi dalla lista degli spettatori.

Il ProjectBrowser fornisce un'interfaccia per visualizzare, creare, cancellare e modificare i dettagli di ogni progetto software (Fig. 3.2).

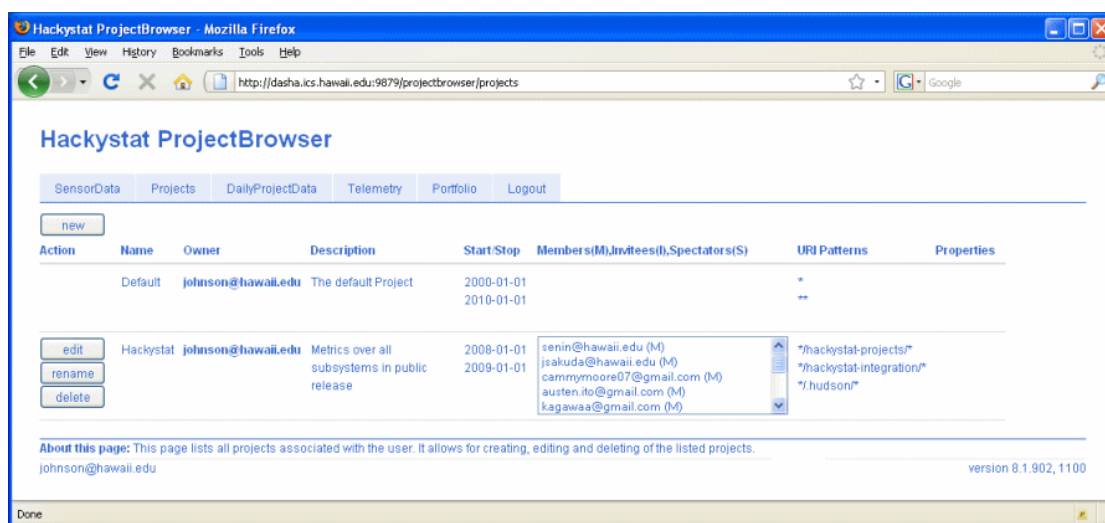


Figura 3.2 – Visualizzazione dei dettagli di progetti software associati all'utente autenticato, tramite ProjectBrowser.

L'osservazione diretta dei sensor data di basso livello raramente permette di investigare a fondo sulle attività di sviluppo e su come migliorarle. Per questo motivo Hakystat fornisce una varietà di astrazioni e analisi su di essi. Un insieme molto utile di astrazioni è chiamato Daily Project Data.

3.3 SERVIZI COMPONENTI

Il sistema hackystat si compone di servizi web RESTful opzionali, che vengono descritti di seguito.

3.3.1 DAILY PROJECT DATA

Un'astrazione particolarmente utile sui dati di basso livello dei sensori, è la Daily Project Data (DPD), che aggrega i dati dei sensori collezionati da tutti i membri di un particolare Progetto in un particolare giorno e per tale progetto. Per esempio può essere utile sapere la quantità totale di Unit Test eseguiti da tutti i membri di un progetto in un giorno. Esempi di applicazioni delle astrazioni DPD includono: uno snapshot sullo stato di un progetto in un singolo giorno; una vista globale sul progetto aggregando le astrazioni DPD rilevate dalla data di creazione del progetto; Software Telemetry che mostra tendenze con livello di granularità giornaliero, settimanale o mensile, utilizzando i dati DPD.

Il servizio DailyProjectData rende tali astrazioni disponibili a tali applicazioni di più alto livello. Hackystat supporta istanze multiple di servizi DPD ma ognuna di esse può essere connessa ad un'unica Sensorbase. Le astrazioni create dal servizio DPD sono le seguenti: build, code issue, commit, complexity, coupling, coverage, devtime, filemetric, issue, unittest.

L'applicazione ProjectBrowser presenta un'interfaccia utente per molte delle DailyProjectData analisi, come mostrato in Figura 3.3 dove è stata richiesta l'analisi di complessità su un progetto Hackystat chiamato "simpletelemetry" dato un singolo giorno. E'anche possibile selezionare più di un progetto contemporaneamente, per effettuare eventuali analisi comparative.



Figura 3.3 – ProjectBrowser mostra il risultato dell’analisi sulla complessità del progetto “simpletelemetry” effettuata sul giorno 22-07-2007.

Le analisi di DailyProjectData possono essere utili e interessanti in sé, ma Hackstat costruisce su tali analisi un livello di astrazione anche più alto, chiamato Software Project Telemetry.

3.3.2 SOFTWARE PROJECT TELEMETRY

I sensori forniscono un modo per collezionare dati di basso livello riguardi processi e prodotti software. Ad esempio un’istanza di sensor data di tipo DevEvent indica che uno sviluppatore ha salvato il file Foo.java alle 11:59:59am del 2007-09-09. Tale informazione è di difficile utilizzo nella gestione dei progetti e in fase di decision making. E’ necessario astrarre in modo da renderla un’informazione significativa.

Il secondo problema consiste nello scegliere astrazioni tali da soddisfare le necessità di differenti organizzazioni che lavorino su differenti processi.

Software Project Telemetry è un meccanismo di analisi per Hackstat che mira a risolvere sia il problema di astrarre sui sensor data, sia il problema di soddisfare differenti necessità degli utenti. Software Project Telemetry è un approccio alla gestione in-process di progetti software basato sulla generazione e l’analisi di misure di processo e prodotto e delle loro tendenze nel tempo. Per rendere i dati di basso livello utili nella gestione dei progetti e nel decision making, supporta la creazione di curve di tendenza che mostrano come differenti caratteristiche dello sviluppo software si

modifichino nel tempo. Per supportare diverse abitudini di lavoro, Telemetry fornisce un linguaggio specifico del dominio che permette la creazione di curve di tendenza personalizzate, chiamate “telemetry stream”, e ne consente la visualizzazione (e quindi la comparazione) all’interno di un unico diagramma. Lo stesso linguaggio definisce un insieme di diagrammi di base forniti, con cui è possibile configurare il servizio di analisi Telemetry. Per esempio il diagramma “ProductQATrends” fornisce una vista delle tendenze sulla qualità di un prodotto includendo i telemetry stream Coverage, invocazioni di Test e Code Issues. I sensor data vengono aggregati a seconda della granularità che si desidera specificare per il diagramma Telemetry, che può equivalere a giorni, settimane o mesi. Dal diagramma risultante è possibile selezionare telemetry stream di interesse (per esempio Code Issues) e richiederne l’inclusione in un grafico. L’applicazione ProjectBrowser fornisce un’interfaccia di alto livello per alcune delle funzionalità offerte da Telemetry (Fig. 3.4).

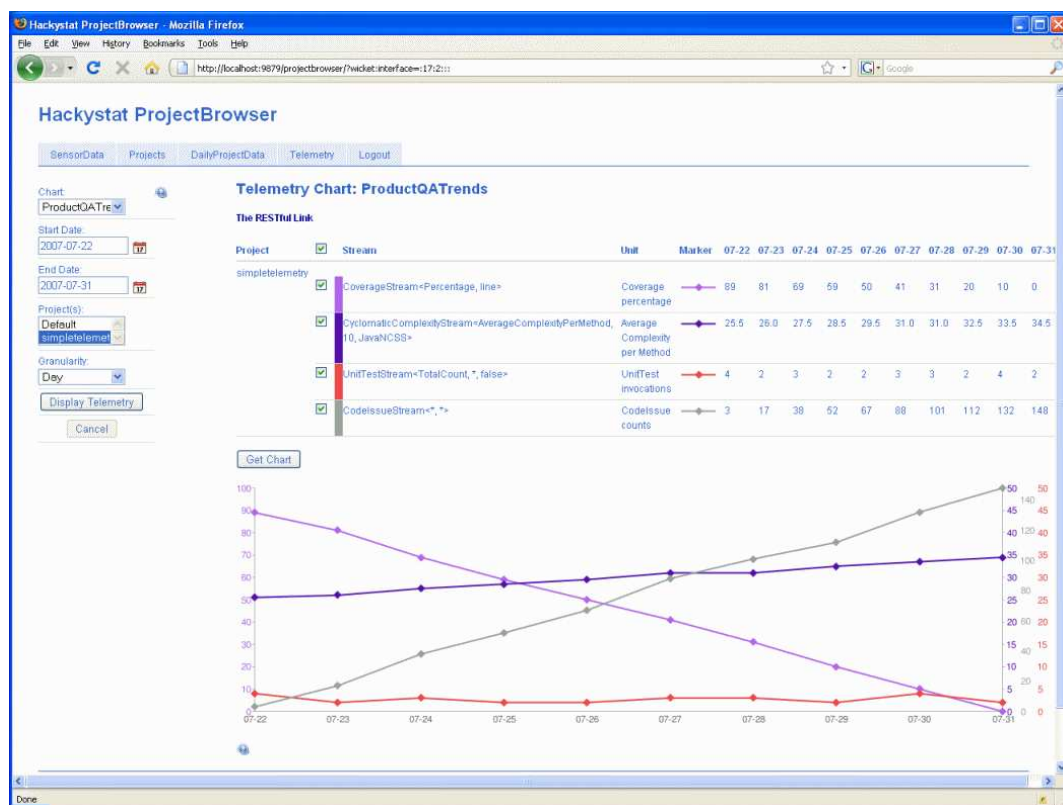


Figura 3.4 – ProjectBrowser mostra un grafico sull’andamento del progetto “simpletelemetry” nel periodo di tempo dal 22-07-2007 al 31-07-2007, rispetto alle misure di qualità test coverage e frequenza di invocazione test (in decremento), complessità e code issues (in incremento).

L’interfaccia permette di selezionare un particolare grafico a scelta, il quale definisce un insieme di linee di tendenza, anche dette “stream” da includere nel grafico. In tal modo è anche possibile scoprire correlazioni tra le diverse misure di qualità. E’ anche necessario specificare un intervallo di

tempo e uno o più progetti contemporaneamente, in caso di eventuali analisi comparative tra progetti diversi.

3.3.3 SOFTWARE PROJECT PORTFOLIO

Tuttavia Telemetry impone un limite al numero di linee di tendenza visualizzabili contemporaneamente nei grafici, in quanto oltre le 10 o 15 linee, i risultati forniti diventano di difficile interpretazione. Ciò rende tale componente inadatta per le organizzazioni che necessitino di confrontare un maggior numero di progetti individuali e i loro andamenti. Inoltre non è fornita un'esplicita indicazione sulla positività o negatività degli andamenti: viene lasciato tutto a conclusioni che l'utente deve ricavare autonomamente; nonostante sia spesso possibile individuare delle soglie universali che, a seconda del contesto, segnalino automaticamente un andamento positivo o meno.

Nei casi in cui il problema principale non consiste solo nella comprensione e gestione di un singolo progetto ma di decine o centinaia di progetti, un "portfolio" di progetti. Collezionare, analizzare e utilizzare dati nei vari progetti aventi differenti caratteristiche e contesti, mantenendo un'accettabile rapporto tra costi e benefici, costituisce una sfida. Un potenziale vantaggio risiede invece nel poter trarre giovamento ogni gruppo di lavoro dall'esperienza o avanzamenti dell'altro. Per esempio un team che abbia verificato quanto un nuovo strumento di sviluppo incrementi significativamente il test coverage, nelle organizzazioni tradizionali deve spendere tempo e risorse per diffondere la notizia agli altri team ("information push"), a vantaggio dell'intera azienda. Utilizzando invece un efficace meccanismo di analisi a livello di portfolio è possibile tradurre la fase di "information push" in "information pull": team interessati ad aumentare il proprio livello di test coverage possono utilizzare le analisi portfolio per trovare altri team che hanno raggiunto il livello desiderato, e richiedere dunque maggiori informazioni sulle ragioni di tale successo.

Il Software Project Portfolio Analysis (SPPA) di Hackystat mira esattamente a supportare le organizzazioni che necessitano di gestire numerosi progetti contemporaneamente, raggiungendo i seguenti obiettivi:

- **Scalabilità:** le analisi portfolio consentono la visualizzazione e il confronto di una elevata quantità di progetti, dalle dozzine alle centinaia.
- **Valutazione:** le analisi portfolio comprendono regole che supportino gli sviluppatori nella comprensione dei risultati, marcando ove possibile i risultati come positivi o negativi, con immediata efficacia visiva.

Grazie alla SPPA pur non conoscendo i dettagli, si è in grado di stabilire immediatamente quali progetti necessitano di maggiore attenzione in quanto riscontrano un più alto numero di problemi, e quali no (Fig. 3.5).

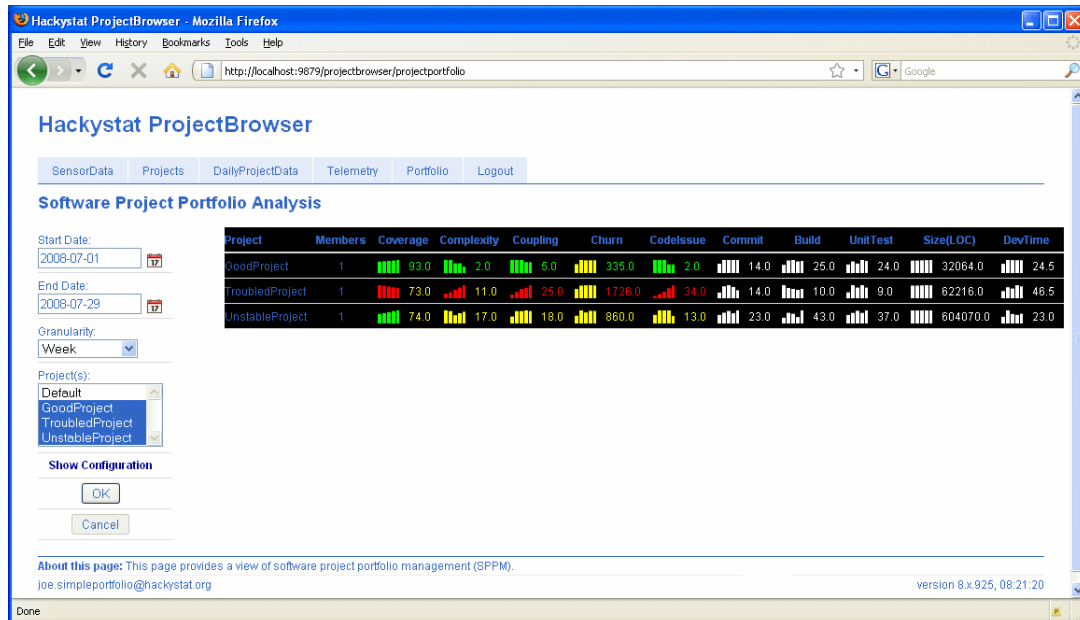


Figura 3.5 – Software Project Portfolio Analysis all’interno dell’interfaccia fornita da ProjectBrowser. Sono comparati tre progetti diversi nel periodo di tempo dall’ 1-07-2008 al 29-07-2008, considerando i sensor data raggruppati per settimane. I colori indicano uno stato di salute: rosso – cattivo; giallo – instabile; verde – buono; bianco – neutrale.

In figura 3.5 notiamo che la valutazione sulla qualità dei progetti è suddivisa in base ai telemetry stream considerati, (Coverage, Complessità o Coupling, etc.) e consiste in un istogramma Sparkline e nel valore numerico più recente. Sia l’istogramma che il valore numerico indicano il trend e sono colorati indipendentemente tra loro, a seconda del superamento o meno dei valori soglia per gli stati di salute neutrale (bianco), cattivo (rosso), instabile (giallo), buono (verde). In particolare lo stato “neutrale” è indispensabile in un ambito in cui non sempre è facile determinare quanto una certa caratteristica sia positiva o meno. Per esempio un’alta frequenza di commit non è necessariamente una tendenza positiva, ma può essere utile apprenderla per individuare possibili cause dei valori raggiunti da altre misurazioni.

Infine SPPA fornisce un tipo di compressione delle informazioni non presente in Telemetry. Per esempio in Figura 3.5 sono stati utilizzati l’equivalente di 30 Telemetry stream (10 per ognuno dei 3 progetti). Di conseguenza è evidente che la quantità di progetti e di stream può salire indefinitamente, senza perdita di usabilità.

CAPITOLO IV

4 HACKYSTAT LINKED SENSOR DATA

Hackystat supporta il monitoraggio e l'analisi della qualità di progetti software e, quindi, indirettamente anche degli sviluppatori. Considerati il futuro prossimo del Web e la tecnologia emergente dei Linked Data è stata ideata un'unione tra le opportunità da essa offerte e i dati che Hackystat rende disponibili riguardo ai progetti e agli sviluppatori, partendo da un concetto: l'esperienza e i problemi affrontati da altri team di sviluppo potrebbero essere di aiuto ad uno sviluppatore qualsiasi, se egli potesse usufruire di efficaci collegamenti tra differenti progetti e sviluppatori particolarmente competenti. Lo sviluppo di software [34] è un processo che richiede tempo e denaro e, malgrado tutto, la qualità del prodotto finale non sempre è garantita. Una porzione considerevole di risorse vengono spesi nel risolvere problemi che sono simili o identici a quelli già risolti da altri in altri progetti

. L'obiettivo che si pone questa tesi, come spiegato nel primo capitolo, consiste nel ridurre i tempi generalmente richiesti dagli sviluppatori per reperire suggerimenti o esempi riguardo l'utilizzo di particolari librerie o tool o linguaggi di programmazione, oppure riguardo particolari scelte implementative, attraverso la creazione dinamica di collegamenti con dati esterni riguardo progetti software simili (che utilizzino stesse librerie o tool) issue simili a quelli incontrati (etichettati allo stesso modo o generati dalla stessa risorsa), e attraverso la ricerca di altri sviluppatori caratterizzati da particolari competenze di interesse per uno sviluppatore che cerchi particolari suggerimenti; dove i collegamenti dinamici creati sono tra Linked Data e le risorse sono descritte tramite RDF. I Linked Data quindi, vengono utilizzati in modo innovativo per la risoluzione di problemi comuni nello sviluppo di software in generale, trasformando lo sviluppo in sviluppo collaborativo, sfruttando Hackystat come sorgenti dati.

Nel prossimo paragrafo viene analizzata l'architettura del Sistema sviluppato per realizzare la soluzione ai problemi proposta, e le sue dinamiche.

4.1 ARCHITETTURA DEL SISTEMA

Hackystat Linked Sensor Data (LiSeD) è la componente creata allo scopo di applicare i Linked Data allo sviluppo collaborativo del Software. Infatti Hackystat, come spiegato nel precedente capitolo, supporta esattamente la gestione dei progetti e il lavoro di gruppo su di essi, e la sua

architettura client/server supporta anche team di sviluppo i cui membri siano fisicamente distanti. L'ambiente di monitoraggio del lavoro è distribuito così come lo è il lavoro di sviluppo stesso. Quindi i dati raccolti da Hackstat sono ideali per lo scopo che ci si è inizialmente posti: collegare i dati di sviluppo (sui progetti, gli issue e gli sviluppatori) di progetti diversi per supportarne la cooperazione indiretta che possa essere d'aiuto ad un qualsiasi sviluppatore nel superamento dei problemi da lui comunemente incontrati.

Poiché, come spiegato, Hackstat ha un'architettura strutturata in componenti semi-indipendenti tra loro e comunicanti tramite REST API, nella creazione del nuovo servizio che fornisce Linked Sensor Data sono stati rispettati tali principi. In altre parole il servizio è una componente opzionale del sistema Hackstat semi-indipendente che, se eseguita, necessita di utilizzare i dati forniti da Software Project Telemetry e SensorBase (i quali devono dunque essere obbligatoriamente installati e in esecuzione insieme a LiSeD).

I metodi di interazione con la componente sono gli stessi previsti per tutti gli altri servizi Hackstat, ovvero tramite browser, REST API e Java client. In aggiunta vi è anche un'interfaccia grafica e, come vedremo, sono esposti anche degli Sparql endpoint che permettono di effettuare ricerche sui linked sensor data (funzionalità ancora non prevista da nessun altro servizio Hackstat).

Un diagramma riassuntivo dell'architettura di LiSeD è mostrato in Figura 4.1.

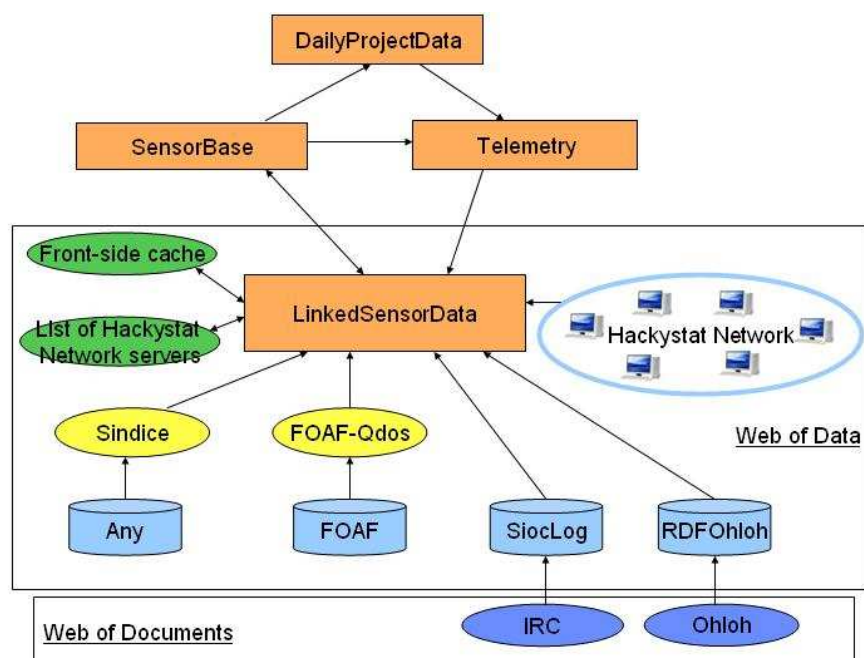


Fig. 4.1 – Overview sull'architettura di Hackstat Linked Sensor Data.

Le componenti di Hackstat coinvolte sono rappresentate dai riquadri arancione; mentre gli ovali in verde rappresentano le uniche risorse che il server LiSeD memorizza in locale. I restanti oggetti

rappresentano tutti i dati esterni collegati con i dati del server LiSeD. I Linked Sensor Data si costruiscono creando collegamenti con i seguenti dati:

- Linked data di altri LiSeD server inclusi nell'Hackystat Network cui si è connessi, filtrati tramite i relativi Sparql endpoint;
- Linked data del dataset (Web of Data) RDFS Ohloh contenente Linked Data estratti dalle pagine (Web of Documents) di Ohloh, filtrati tramite Ohloh API;
- dataset SiocLog contenente Linked Data estratti dai log di alcuni channel IRC;
- dataset di profili FOAF filtrati tramite il motore di ricerca semantico FOAF-Qdos;
- dataset qualsiasi filtrati tramite il motore di ricerca semantico Sindice.

Le frecce rappresentano i flussi di dati scambiati: le uniche occasioni in cui lo stesso LiSeD fornisce le sue informazioni all'esterno sono durante la comunicazione con la SensorBase e con le risorse memorizzate in locale. In tali occasioni si comunicano eventuali modifiche effettuate dagli utenti sui dati originari rispettivamente della SensorBase o della cache o dell'elenco di Hackystat Network LiSeD server esterni.

Il LinkedSensorData è un server ma se si desidera interagire tramite interfaccia grafica è fornita una GUI lato client inclusa nel progetto. Non esiste invece una componente client per i servizi Hackystat SensorBase, DailyProjectData e Telemetry: per Hackystat le uniche componenti lato client sono i sensori e i servizi di interfaccia utente quali ProjectBrowser e Tickertape.

Di seguito verranno approfondite le tematiche riguardanti, nell'ordine, le risorse Hackystat utilizzate (dettagli sui riquadri arancine in Fig. 4.1); la Rete Hackystat (ovale azzurro in Fig. 4.1); le informazioni aggiunte dal servizio LiSeD (dettagli su LiSeD stesso in Fig. 4.1); la struttura dei Linked Data creati, la modalità di collegamento con dataset esterni dettagli sui cilindri azzurri in Fig. 4.1); il sistema di caching (dettagli sugli ovali verdi in Fig. 4.1).

4.2 DATI HACKYSTAT UTILIZZATI

I principi architetturali REST prevedono il concetto di risorsa identificata univocamente da un URI e le URI utilizzate dai servizi REST Hackystat sono http. Ricordando i principi dei Linked Data che sono:

1. URI per identificare risorse informative e non
2. URI http
3. URI visitabili tramite browser, e che forniscano informazioni utili sulla relativa risorsa
4. Collegamenti RDF con altre URI

Risulta che i primi due principi sono soddisfatti dai dati Hackystat anche senza l'intervento di LiSeD, ognuno associato ad URI http. Tuttavia si è reso necessario l'utilizzo di URI differenti in quanto i precedenti non erano effettivamente riferiti a Linked Data, ovvero non erano riferiti a risorse descritte in RDF né tanto meno contenenti collegamenti RDF ad altre risorse. LiSed invece, crea Linked Data a partire dai dati dei sensori (Linked Sensor Data) prelevati tramite SensorBase, o dalle loro astrazioni prelevate tramite Telemetry. Di seguito elencheremo quali dati vengono prelevati da SensorBase e quali da Telemetry, e per quale scopo.

La SensorBase memorizza i dati dei sensori su cui è possibile ottenere delle “viste” (logiche e non fisiche) in base alle tipologie in cui i sensor data rientrano. Tali tipologie sono: progetti e utenti. Tramite ProjectBrowser e i servizi Hackystat di più alto livello, si fa riferimento esplicito solo alla vista sui dati di tipo progetto: è sui dati di tipo Progetto che vengono effettuate le analisi DPD, Telemetry e Portfolio, ed è solo per i profili Progetto che esiste una scheda dedicata nel ProjectBrowser. D'altronde tale tipologia è la più dettagliata: contiene un maggior numero di informazioni, quali:

Nome	Descrizione	Esempio
Name	Nome del Progetto	Jupiter-3.3
Description	Descrizione testuale del Progetto	Release 3.3 del plugin di revisione codice Jupiter per Eclipse.
StartTime	Timestamp che indica l'inizio del Progetto	2006-01-31T00:00:00.000
EndTime	Timestamp che indica la fine del Progetto.	2007-01-20T00:00:00.000
Owner	URI corrispondente e al	In caso di GET: http://dasha.ics.hawaii.edu:9876/sensorbase/users/johnson@hawaii.edu

	proprietario di questo Progetto.	In caso di PUT: johnson@hawaii.edu
Members	Insieme di URI di utenti membri di questo Progetto.	In caso di GET: {http://dasha.ics.hawaii.edu:9876/sensorbase/users/johnson@hawaii.edu, http://dasha.ics.hawaii.edu:9876/sensorbase/users/hongbing@hawaii.edu} In caso di PUT: {johnson@hawaii.edu, hongbing@hawaii.edu}
Invitations	Insieme di URI di utenti invitati a questo Progetto.	In caso di GET: {http://dasha.ics.hawaii.edu:9876/sensorbase/users/johnson@hawaii.edu, http://dasha.ics.hawaii.edu:9876/sensorbase/users/hongbing@hawaii.edu} In caso di PUT: {johnson@hawaii.edu, hongbing@hawaii.edu}
Spectators	Insieme di URI di utenti spettatori di questo Progetto.	In caso di GET: {http://dasha.ics.hawaii.edu:9876/sensorbase/users/johnson@hawaii.edu, http://dasha.ics.hawaii.edu:9876/sensorbase/users/hongbing@hawaii.edu} In caso di PUT: {johnson@hawaii.edu, hongbing@hawaii.edu}
UriPatterns	Insieme di URI Patterns	{file:/*/Jupiter/*}
Properties	Insieme di coppie di stringhe chiave-valore contenenti meta-data sul Progetto.	{TDD-Group=true}

Mentre i dati di tipo Utente includono solo quattro campi:

Nome	Descrizione	Esempio
Email	Indirizzo e-mail dell'utente	qzhang@hawaii.edu
Password	Password dell'utente	changeMe1234
Role	Tipo di utente ("admin" oppure "basic")	basic
Properties	Insieme di coppie di stringhe chiave-valore contenenti meta-data sull'utente.	{DesktopPrefs=Google}

Nel corso di questa tesi, le informazioni contenute nei tipi di dato Progetto e Utente sono state estese e affiancate dal nuovo tipo di dato "Issue" sul quale non si hanno precise indicazioni essendo in corso di sviluppo e ancora privo di un'ufficiale descrizione.

4.3 RETE HACKYSTAT

Hackystat pone delle limitazioni all'accesso ai propri dati secondo una strategia di tutela totale della privacy degli utenti dai cui sensori sono stati generati tali dati. Gli utenti hanno i permessi necessari a:

- visualizzare nei dettagli unicamente i dati generati dai propri sensori. Di conseguenza utenti non registrati su Hackystat, a cui quindi non è associato alcun sensor data, non hanno accesso ad alcun dato.
- Visualizzare risultati solo delle astrazioni (DPD, Telemetry, etc.) associate a progetti di cui si è membri o spettatori. Tali astrazioni vengono costruite processando i sensor data non solo dell'utente correntemente autenticato, bensì di tutti i membri del progetto. Tuttavia i dettagli dei sensor data degli altri membri del progetto vengono astratti e i loro dettagli continuano ad essere visibili unicamente ai relativi proprietari.

Solo l'amministratore ha accesso ai sensor data di tutti gli utenti e alle astrazioni associate a qualsiasi progetto. Il motivo di tali limiti risiede nel tipo di dati memorizzati da Hackystat, i quali sono particolarmente delicati dal punto di vista della privacy. Tuttavia uno sviluppatore che tema di rendere pubblici informazioni riguardanti la propria attività di sviluppo, ha evidentemente abitudini negative da nascondere; altrimenti pubblicarli potrebbe al contrario diventare motivo di orgoglio.

Al di là delle considerazioni personali che è possibile avanzare sulla tipologia di utenti che preferirebbero conservare privati o rendere pubblici i propri dati dei sensori, è certamente realistico implementare un sistema di salvaguardia privacy più flessibile, personalizzabile dagli utenti stessi. L'aggiunta della funzionalità di personalizzazione ai livelli di privacy sui dati deve però avvenire nella SensorBase e, quindi, non è di mia personale competenza e qualunque modificata eventuale da apportare dev'essere preventivamente discussa con gli altri membri del progetto Hackystat. Lo stesso Hackystat può trarre beneficio dalla liberazione almeno parziale dei propri dati, come dimostrato nel paragrafo "superamento dei data silos: vantaggi" nel capitolo II. Comunque in previsione di un simile sviluppo futuro i Linked Data creati includono anche la meta-descrizione di un campo-flag che specifichi tramite valore numerico, il livello di privacy stabilito dall'utente. Tale valore nonostante attualmente non sia ancora supportato dalla SensorBase, è anche modificabile tramite LinkedSensorData-GUI.

Queste limitazioni sull'accesso però, rendono inutilizzabile il servizio di ricerca che si intende fornire sui Linked Data, se non all'amministratore. Per ovviare questo problema è stata pianificata la seguente strategia. Si serializza un elenco dei server Hackystat (ognuno associato alle credenziali di autenticazione del relativo amministratore) che desiderano rendere pubblici tutti i loro dati: verso tali server vengono propagate le ricerche richieste dagli utenti, e i dati di ognuno di essi saranno tutti accessibili sfruttando appunto, le credenziali dell'amministratore.

Per tutelare la password dell'amministratore, essa una volta aggiunta all'elenco non viene mai comunicata né in chiaro né criptata. Non sono forniti servizi che permettano di risalire alle password degli amministratori. Gli unici dettagli che è possibile ottenere sono le URI dei server e le e-mail degli amministratori.

4.3.1 ALGORITMO UTILIZZATO

L'algoritmo implementato per la creazione di una siffatta rete di server Hackystat è il seguente.

Quando un utente inoltra una query verso uno Sparql endpoint:

1. si controlla se unitamente alla query ricevuta è stato allegato l'elenco XML dei server della rete Hackystat
 - a. in caso affermativo:
 - i. si tratta dell'elenco cosiddetto "noLoop" ovvero utile ad evitare la creazione di cicli infiniti, segnalando quali server sono già stati interrogati e non necessitano di essere contattati nuovamente dal server locale per la

- propagazione della ricerca. L'elenco noLoop viene momentaneamente conservato fino al termine della propagazione della ricerca;
2. la query viene eseguita in locale solo sui dati accessibili in base ai permessi associati all'utente (nessun dato se non ci si è autenticati; solo i propri se ci si è autenticati; tutti se ci si è autenticati come amministratori);
 3. Per ogni server X incluso nell'elenco locale
 - a. Se il server X non è compreso anche nell'elenco "noLoop" allora
 - i. La query viene inoltrata allo Sparql endpoint del server X
 - ii. Si attende una risposta
 - iii. Si integrano gli URI ottenuti in risposta, alla risposta finale
 4. L'elenco noLoop viene cancellato;
 5. Si restituisce all'utente una risposta (nel formato XML secondo le raccomandazioni W3C riguardo i risultati di query Sparql [29]) in cui sono state incluse le risposte ottenute da tutti i server della rete Hackystat grazie alla propagazione della ricerca.

4.4 LINKED SENSOR DATA

Hackystat fornisce le informazioni utili al raggiungimento degli obiettivi posti tramite la costruzione, su di essi, dei Linked Data. I dati di base considerati sono stati già descritti nei paragrafi precedenti, ma essi oltre ad essere stati meta-descritti con RDF sono stati anche integrati con informazioni aggiuntive opzionali. Tali dettagli aggiuntivi sono potenzialmente utili a migliorare la capacità del sistema di individuare risorse esterne collegabili perché correlate: la correlazione può così essere misurata in base ad un maggior numero di dettagli e quindi essere meno generica.

I metadata vengono generati dinamicamente, su richiesta, in quanto l'utilizzo di un qualsiasi supporto di memorizzazione è stato sconsigliato dalla comunità di sviluppatori Hackystat, per non appesantire il sistema. Tuttavia esso avrebbe apportato di certo miglioramenti prestazionali.

L'implementazione di collegamenti completamente dinamici con dataset esterni qualsiasi tramite semantic search engine, seguendo la strategia descritta in precedenza per superarne i limiti grazie all'utilizzo di void e Umbel, non è stata ancora completata. Sebbene siano stati associati, tramite Semantic Web crawling Sitemap Extension, al dataset i suoi metadata in cui sono stati utilizzati void e i subject concept di Umbel, non è stata implementata la parte del progetto responsabile di costruzione e inoltro di un'interrogazione appropriata al semantic search engine.

Quindi attualmente le ricerche di risorse correlabili con i Linked Sensor data vengono indirizzare a specifici dataset quali: RDFOhloh, risorse FOAF, SiocLog e Hackystat dataset dei LiSeD server inclusi nella locale Rete Hackystat.

Risorse o utenti esterni possono effettuare delle ricerche tra i dati Hackystat sfruttando uno dei tre Sparql endpoint esposti, a seconda che si desideri ricercare Linked Data di profili progetto, issue o sviluppatore.

4.4.1 LINKED DATA DEGLI SVILUPPATORI

Le descrizioni riguardanti gli sviluppatori sfruttano

- i dati contenuti nella vista “utenti” sui sensor data della SensorBase
 - e-mail, proprietà (di base l’elenco di proprietà è vuoto);
- i dati relativi ai valori di qualità collezionati dall’utente in tutti i progetti in cui è/è stato coinvolto (tranne il progetto “Default”) in un range temporale non maggiore dell’inizio dell’anno corrente;
- un valore di qualità relativo alle capacità comunicative dell’utente, rilevato sfruttando il collegamento con i Linked Data di SiocLog [35] che forniscono informazioni riguardo la quantità di interventi effettuati su alcuni canali IRC;
- Un valore cosiddetto “karma” avente la funzione di riassumere la bravura generale dell’utente in base alle misure di qualità collezionate;
- Nome e cognome; homepage, weblog, nickname, specificabili opzionalmente dall’utente stesso tramite la LinkedSensorData-GUI.

Le misure di qualità, sia quelle ricavate tramite Telemetry sia quelle ricavate tramite i Linked data di SiocLog, non sono frutto di indagini approfondite né di calcoli complessi, e necessitano di essere approfondite per fornire un quadro realistico, efficace, delle reali competenze di uno sviluppatore.

Infatti i semplici calcoli effettuati sono i seguenti:

- Amount of Code Issues
 - Media del numero di code issues
- Build quality
 - Rapporto tra numero Build terminati con successo e Build falliti
- Commit frequency
 - Media del numero di Commit effettuati

- Development time
 - Media del tempo dedicato attivamente allo sviluppo SW
- Unit Test quality
 - Rapporto tra Unit Test terminati con successo e Unit Test falliti
- Communication effort
 - Totale messaggi inviati nei canali IRC monitorati da SIOCLog
- Karma
 - Somma di tutti i valori ottenuti dai calcoli appena elencati.

Il vocabolario utilizzato per specificare i dettagli dell'utente/sviluppatore è FOAF, mentre non sono stati trovati vocabolari pre-esistenti per descrivere le appena elencate misure di qualità dell'attività di sviluppo software. E' stato quindi creato un piccolo vocabolario Hackstat con l'unico scopo di sopperire queste poche mancanze (Fig. 4.4).

4.4.1.1 COLLEGAMENTO CON RISORSE ESTERNE

I Linked Data sullo sviluppatore comprendono link RDF ad eventuali ulteriori profili (in forma di Linked Data) esterni associati allo stesso sviluppatore. Tali profili vengono cercati tra

- I profili FOAF utilizzando Foaf-Qdos prima e, in caso non venga trovato alcun risultato, Sindice. Questo iter viene seguito in quanto Foaf-Qdos è un motore di ricerca semantico specializzato su profili FOAF più efficace di Sindice, ma tramite il quale spesso non si ottengono risultati a causa della ridotta quantità di risorse indicizzate su cui effettua le ricerche dei profili. Si creano RDF link di tipo
 - sameAs in caso sia specificato lo stesso indirizzo e-mail e/o homepage e/o weblog e/o nome e/o cognome in due risorse diverse;
- I profili SiocLog, solo nel caso in cui venga fornito un nickname. Solo in tal caso infatti è possibile risalire ad un URI SiocLog probabilmente esistente del tipo `http://irc.sioc-project.org/users/[nickname]#user`. Non esistono altri modi di ricercare dati correlabili in quanto non è stato esposto alcuno Sparql endpoint. Si creano RDF link di tipo
 - seeAlso
- I profili RDFOhloh. Anche in questo caso non viene fornito alcuno Sparql endpoint tramite cui effettuare ricerche di dati correlabili ai propri. Ci si limita quindi ad effettuare ricerche tramite la Ohloh API (con i limiti tipici dei motori di ricerca in stile Web of Documents), estrarre dai risultati del tipo `http://www.ohloh.net/accounts/[ID]` i vari identificatori ID dei

profili, ed inserirli in altrettante URI di RDFOhloh del tipo:

[http://rdfohloh.wikier.org/user/\[ID\]](http://rdfohloh.wikier.org/user/[ID]) . Si creano RDF link di tipo

- sameAs in caso sia specificato lo stesso indirizzo e-mail in due risorse diverse;
- seeAlso in caso sia specificato lo stesso nickname in due risorse diverse;
- I profili Hackystat esterni effettuando query sugli Sparql endpoint dei server LiSeD coinvolti nella locale rete Hackystat. Si creano RDF link di tipo
 - sameAs in caso sia specificato lo stesso indirizzo e-mail in due risorse diverse;
 - seeAlso in caso sia specificato lo stesso nickname e/o homepage e/o weblog e/o nome e/o cognome in due risorse diverse;

4.4.1.1.1 IL SERVIZIO FOAF-QDOS

FOAF (Friend Of A Friend) è un vocabolario utile a descrivere le persone, le loro attività e relazioni con altre persone e oggetti. E' lo strumento per la creazione di social network decentralizzate e libere, senza alcun bisogno di una base di dati centrale, completamente controllate dai singoli utenti e non da una singola organizzazione. Il progetto creato nel 2000 da L. Miller e D. Brickley, è associato alla sfida riguardo una sua maggiore accessibilità, raccolta da Garlik fornendo il servizio Foaf-Qdos che permette la ricerca e la navigazione tra milioni di file FOAF. Il servizio mira ad indicizzare l'intero universo di file FOAF, tuttavia la quantità di file indicizzati ancora non è molto estesa. Per questo motivo spesso si rende necessaria, con successo, la ripetizione della ricerca tramite Indice.

Il vantaggio principale di Foaf-qdos consiste nella possibilità di effettuare reverse search, e trovare chi fa riferimento ad un dato URI [36]. Inoltre è possibile ottenere ottimi risultati specificando anche solo una delle Inverse Functional Properties quali foaf:mbox, foaf:mbox_sha1sum, foaf:homepage o foaf:weblog e cercando i profili di chi dichiara di conoscere qualcuno i cui URI di homepage e/o di weblog e/o di e-mail corrispondono a quelli inseriti.

4.4.1.1.2 IL SERVIZIO SIOCLOG

L'applicazione software Sioclog supporta il social networking e la collaborazione online attraverso l'osservazione delle discussioni su alcuni pubblici canali IRC, archiviando i log delle discussioni successivamente pubblicati come Linked Data sul Web utilizzando il vocabolario SIOC. I partecipanti possono successivamente fornire il proprio Web ID o personale profilo FOAF e il profilo FOAF conferma di ritorno tale collegamento, garantendone l'autenticità.

Tale progetto è un inizio di collegamento tra IRC e il Web of data, che si estenderà in futuro per coprire un maggior numero di canali IRC e collezionare più dettagli e collegamenti nei metadata associati ad ogni utente.

4.4.1.1.3 IL SERVIZIO RDFSLOH

Ohloh è un'open source directory che espone una RESTful API mentre RDFSloh [37] è un wrapper di Ohloh che sfrutta tale API (subendone quindi la limitazione di non più di 1000 richieste al giorno inoltrabili) per costruire Linked Data collegati a concetti provenienti da DBpedia e a progetti provenienti da DOAPspace (inaccessibile in realtà da circa un anno).

4.4.2 LINKED DATA DEI PROGETTI

Le descrizioni riguardanti i progetti sfruttano

- I dati contenuti nella vista “progetti” sui sensor data della SensorBase
 - Nome progetto, Responsabile, Data inizio, Data fine, Data ultima modifica, Lista membri, Descrizione
- I dati relativi ai valori di qualità collezionati dai membri del progetto in un range temporale che parte dalla data di nascita del progetto alla data attuale;
- Lista Wiki URI, Lista mirrors URI, Lista linguaggi di programmazione URI, Lista Sistemi Operativi URI, Lista DB di Bug URI, Lista tool e librerie URI, Lista tag URI, Lista fasi di sviluppo URI, Lista ruoli di ogni membro URI, specificabili opzionalmente dall'utente stesso tramite LinkedSensordata-GUI.

Le misure di qualità ricavate tramite Telemetry, non sono frutto di indagini approfondite né di calcoli complessi, e necessitano di essere approfondite per fornire un quadro realistico, efficace, del reale valore di un progetto. Infatti i semplici calcoli effettuati sono i seguenti:

- Amount of Code Issues
 - Media del numero di code issues
- Build quality
 - Rapporto tra numero Build terminati con successo e Build falliti
- Commit frequency
 - Media del numero di Commit effettuati
- Development time

- Media del tempo dedicato attivamente allo sviluppo SW
- Unit Test quality
 - Rapporto tra Unit Test terminati con successo e Unit Test falliti
- Coverage quality
 - Percentuale di classi coperte dai test effettuati

Il vocabolario utilizzato per specificare i dettagli dell'utente/sviluppatore è DOAP, mentre non sono stati trovati vocabolari pre-esistenti per descrivere le appena elencate misure di qualità del progetto software. Anche per questo tipo di mancanza è stato creato il vocabolario Hackstat accennato nella precedente sezione (Fig. 4.3).

4.4.2.1 COLLEGAMENTO CON RISORSE ESTERNE

I Linked Data sul progetto software comprendono link RDF ad eventuali ulteriori profili progetto (in forma di Linked Data) esterni relazionabili al corrente. Tali profili vengono cercati tra

- I profili progetto RDFOhloh. Si creano RDF link del tipo
 - seeAlso in caso sia specificato stesso nome e/o tag e/o descrizione in due risorse diverse. Il principio con cui vengono rilevate le risorse progetto da RDFOhloh correlabili, è lo stesso adottato per le risorse utente correlabili ai profili sviluppatore. Per questo motivo nel caso dei progetti, non è possibile effettuare ricerche specifiche sui tool utilizzati.
- I profili progetto Hackstat. Si creano RDF link del tipo
 - sameAs in caso sia specificato lo stesso nome in due risorse diverse
 - seeAlso in caso sia specificato stesso tag e/o descrizione e/o tool in due risorse diverse

4.4.3 LINKED DATA DEGLI ISSUE

Le descrizioni riguardanti gli issue sfruttano

- Sensor data di tipo “Issue” provenienti dalla SensorBase
 - URI della risorsa da cui l'Issue è stata originata
 - URI dell'Issue Tracking System
 - E-mail dell'utente che ha segnalato l'Issue
 - E-mail dell'utente cui ne è stata affidata la risoluzione

- ID
- Stato
- Priorità
- Milestone
- URI di Issue duplicati e tag, entrambi specificabili opzionalmente dall'utente stesso tramite LinkedSensorData-GUI.

Si tenta anche di automatizzare l'individuazione dei duplicati, individuandoli tramite confronti tra URI che puntano ad uno stesso issue all'interno di un Issue Tracking System.

Il vocabolario utilizzato per descrivere gli issue è Baetle.

4.4.3.1 COLLEGAMENTO CON RISORSE ESTERNE

I Linked Data sugli issue comprendono link RDF ad eventuali ulteriori profili issue (in forma di Linked Data) esterni relazionabili al corrente. Tali profili vengono cercati tra

- I profili issue Hackstat. Si creano RDF link del tipo
 - sameAs in caso siano stati segnalati duplicati manualmente o in automatico, come descritto in precedenza.
 - seeAlso in caso sia specificato stesso tag in due risorse diverse

Non è stato possibile impostare la ricerca di risorse correlabili provenienti da ulteriori dataset in quanto nessuno dei unici due bug archive pubblicati come Linked Data (Helios e Pear) espongono Sparql endpoint tramite cui effettuare ricerche, né i relativi archivi originali o altri archivi quali ad esempio Launchpad, espongono API che permettano la ricerca di issue in base ai tag o agli URI duplicati associati. In particolare la RESTful API di launchpad è in versione beta e probabilmente supporterà tale funzionalità in futuro, in quanto già permette la specifica di tag e duplicati, sebbene i dati pubblicati non sono in forma di Linked Data.

I Linked Data sono una tecnologia tuttora in fase di diffusione ed è certo (grazie alla facilità di creazione/traduzione di dataset dal Web of Documents al Web of Data) un aumento dei dataset di archivi di bug correlabili in futuro.

4.4.4 LINKED DATA DI RISORSE A LIBERO ACCESSO

Alcune risorse non presentano restrizioni sull'accesso unicamente perché non provengono dalla SensorBase. Vengono create da LiSeD in quanto coinvolte nei dettagli descrittivi di progetti, sviluppatori e issue, e sono:

- Descrizioni su Linguaggi di Programmazione
- Descrizioni Sistemi Operativi
- Descrizioni Tool di sviluppo
- Descrizioni Calcolatori

Contenenti unicamente RDF link a metadata esterni ricercati tramite il motore di ricerca Sindice, allo scopo di fornire, su richiesta dell'utente, ulteriori informazioni al riguardo.

4.5 VOCABOLARIO HACKYSTAT

Il vocabolario Hackystat (Fig. 4.2) è stato creato unicamente perché è sorta la necessità di esprimere misure di qualità relative a sviluppatori e progetti SW che non erano definite in alcun altro vocabolario; nonostante vocabolari mirati a descrivere progetti e utenti esistano. Proporre l'aggiunta delle necessarie relazioni ai suddetti vocabolari già esistenti e attendere l'accettazione della proposta da parte delle relative comunità, non è stata un'ipotesi considerabile per motivi logistici di tempo. Di conseguenza è stato creato un vocabolario ad-hoc. Il nome assegnatogli è associato all'abbreviazione "hacky".

In aggiunta alle relazioni più strettamente necessarie, il vocabolario comprende anche alcune classi astratte mirate a supportare funzionalità di ricerca future sulle risorse, in maniera più confortevole.

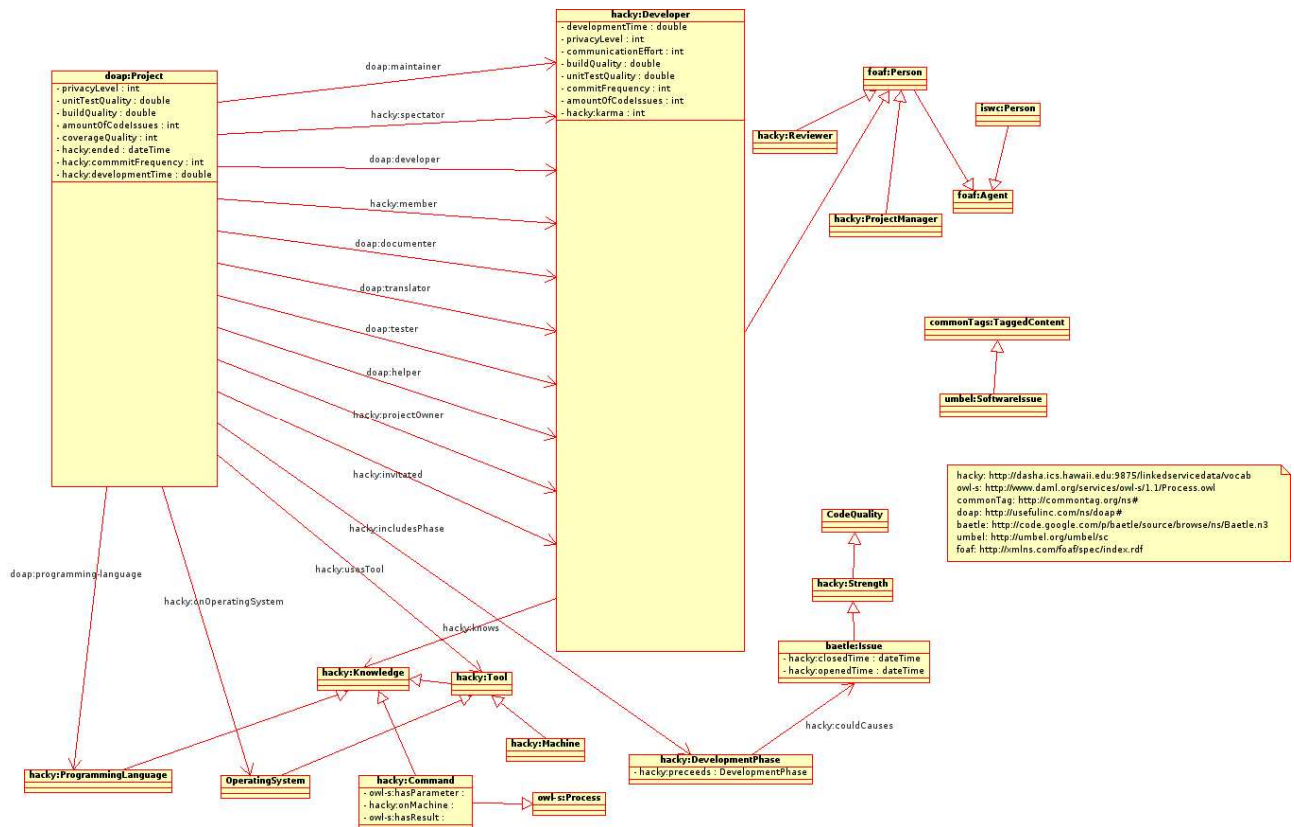


Fig. 4.2 – RDF Schema del vocabolario Hackstat. Nelle immagini seguenti vengono visualizzate sezioni di tale schema altrimenti illeggibili nell'immagine corrente per problemi di spazio.

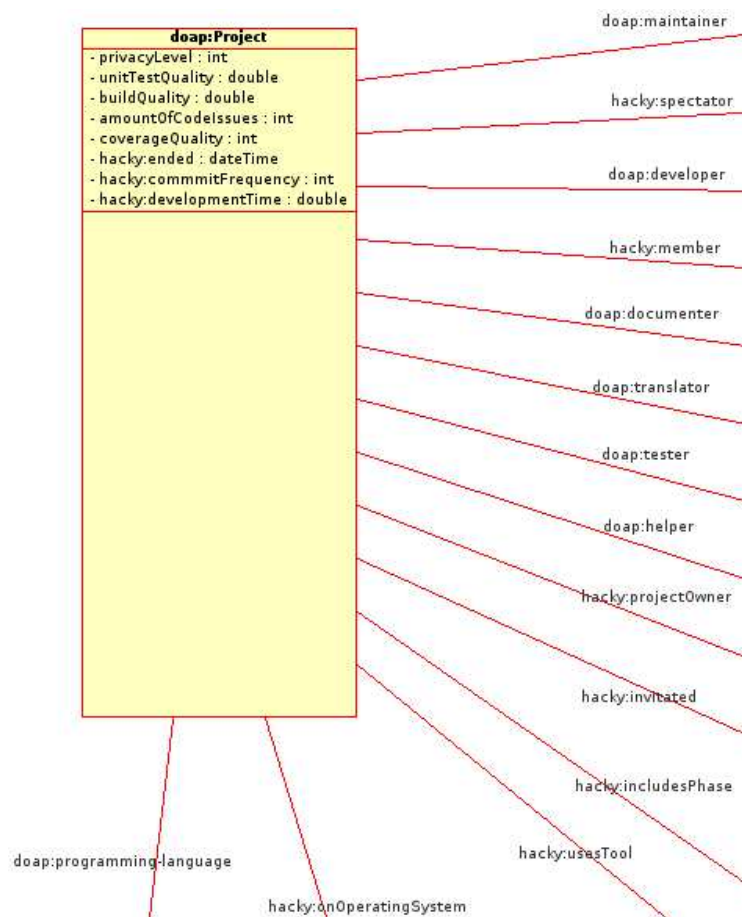


Fig. 4.3 – Particolare dell’RDF schema del vocabolario Hackystat riguardante la risorsa progetto.

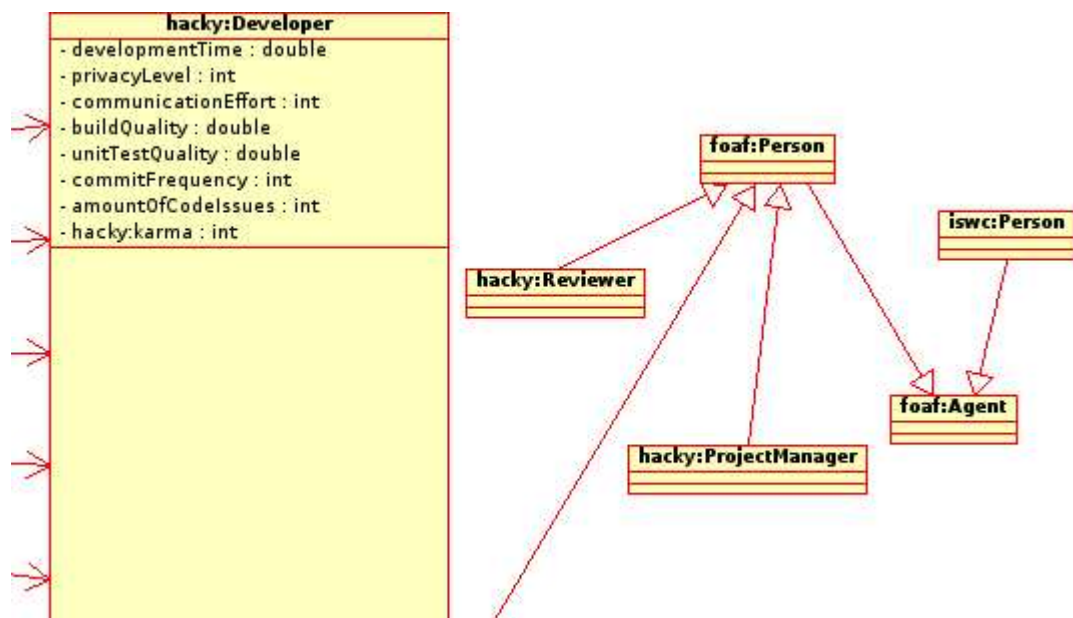


Fig. 4.4 – Particolare dell’RDF schema del vocabolario Hackystat riguardante la risorsa sviluppatore.

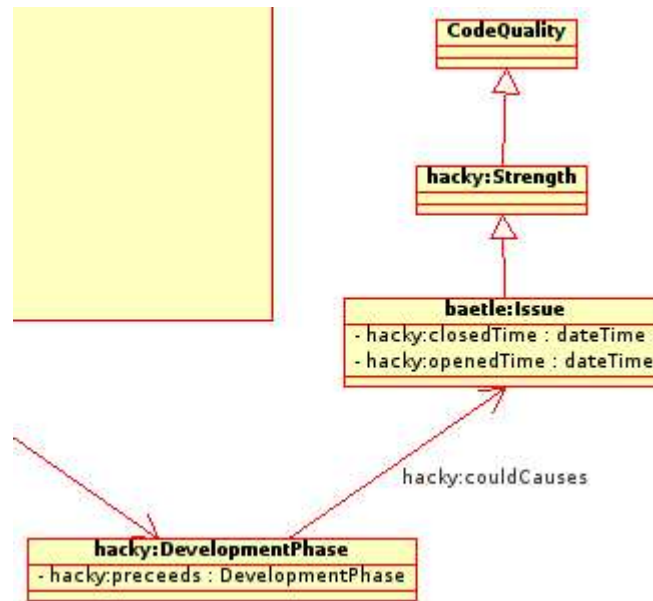


Fig. 4.5 – Particolare dell’RDF schema del vocabolario Hackystat riguardante le risorse issue e fase di sviluppo software.

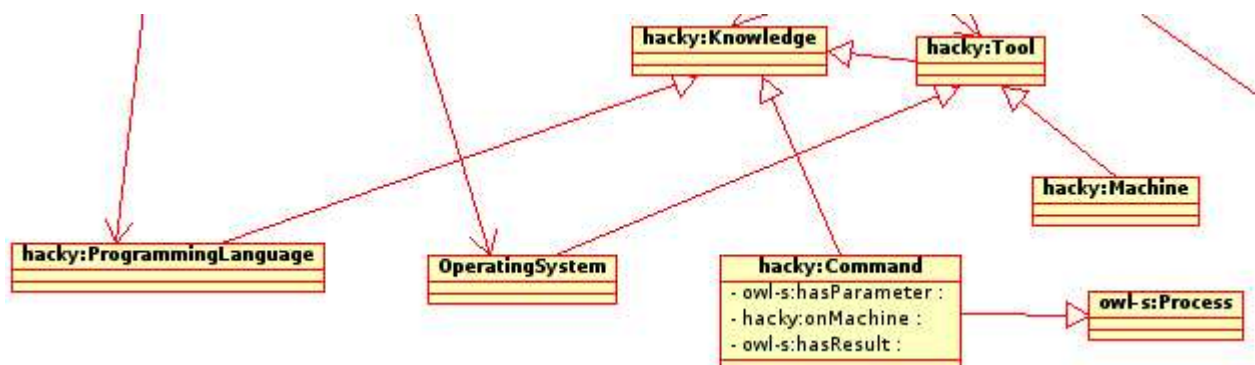


Fig. 4.6 - Particolare dell’RDF schema del vocabolario Hackystat riguardante le risorse a libero accesso.

Per concludere tutti i vocabolari utilizzati nel progetto LiSeD sono elencati in Fig. 4.7.

```

hacky: http://dasha.ics.hawaii.edu:9875/linked servicedata/vocab
owl-s: http://www.daml.org/services/owl-s/1.1/Process.owl
commonTag: http://commontag.org/ns#
doap: http://usefulinc.com/ns/doap#
baetle: http://code.google.com/p/baetle/source/browse/ns/Baetle.n3
umbel: http://umbel.org/umbel/sc
foaf: http://xmlns.com/foaf/spec/index.rdf

```

Fig. 4.7 – Elenco di tutti i vocabolari utilizzati nel progetto LiSeD.

CAPITOLO V

5 IMPLEMENTAZIONE DI LISED

La componente di Hackystat creata, denominata LinkedSensorData, è stata implementata coerentemente con le altre pre-esistenti componenti Hackystat, caratterizzate tutte ad esempio, da un'architettura di tipo REST. Per molti aspetti quali creazione di Linked Data sui Bug e sui progetti, non esistendo esempi precedenti, sono stati utilizzati algoritmi ad-hoc che verranno descritti in questo capitolo, insieme ad altre scelte progettuali.

L'implementazione del RESTful servizio Web Hackystat LinkedSensordata (LiSeD) e dell'applicazione standalone LinkedSensorData_GUI che fornisce un'interfaccia grafica lato client per migliorare l'interazione con l'utente, sono state realizzate in Java, utilizzando anche la libreria Jena che fornisce supporto alla manipolazione di metadata RDF.

5.1 INTERAZIONE COL MODULO

Sono disponibili tre diverse modalità di interazione col modulo LiSeD: tramite browser, tramite Java client e tramite interfaccia grafica. Attualmente non esiste ancora un server LiSeD pubblico e quindi si farà riferimento ad un server LiSeD eseguito da privati.

5.1.1 UTILIZZO DEL BROWSER

LiSeD espone una REST API che permette di

- Aggiungere o cancellare server LiSeD inclusi nella propria Rete Hackystat
- Cancellare dati dalle cache
- Esplorare i dati. Per esempio visualizzare un elenco di utenti, progetti, issue, oppure i loro profili
- Cercare utenti, progetti e issue all'interno della propria Rete Hackystat da privati.

Tutti i LinkedData sono accessibili quindi tramite richieste HTTP GET rendendo sufficiente inserire nel proprio browser l'URI corretto per visualizzare ciò che si desidera. Tuttavia nel caso si volessero utilizzare altri metodi oltre a GET, quali POST o DELETE entrambi egualmente disponibili nell'API, oppure si desidera specificare il formato preferito in risposta (specificando cioè un preferito linguaggio di serializzazione RDF), si consiglia di utilizzare il Firefox Poster Addin per utenti Firefox, oppure un plugin equivalente per qualsiasi altro browser, che permetta di inoltrare

richieste in cui si possa specificare l'Action della richiesta, il Content/type ed eventualmente allegare anche un payload (Fig. 5.1).

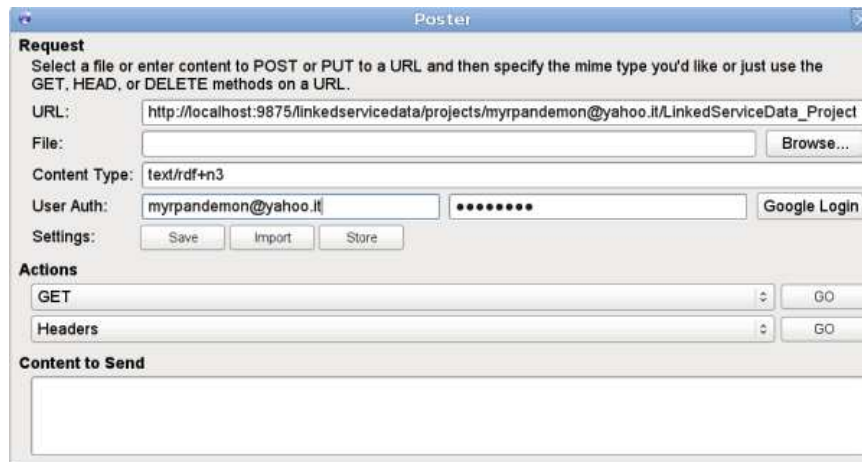


Figura 5.1 – Firefox Poster Addin interface, in cui sono stati specificati un URL, un'azione e un Content-type preferito, nonché le credenziali per l'autenticazione (in caso di accesso a risorse aventi restrizioni sull'accesso)

Un esempio di risposta a tale richiesta è in Figura 5.2.

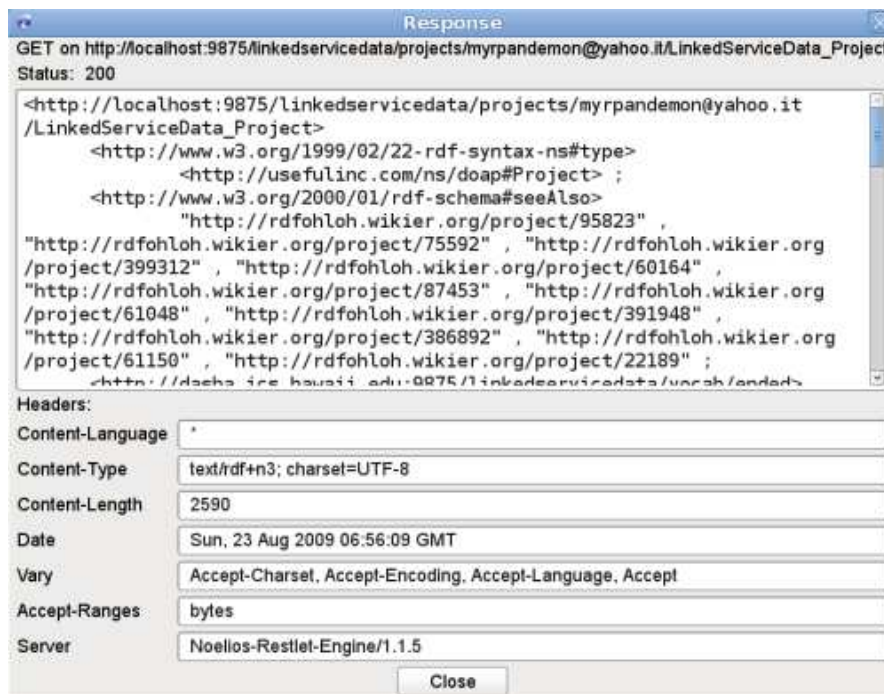


Figura 5.2 – Risposta ottenuta dalla REST API del LiSeD server tramite il Firefox Poster Addin.

LiSeD per funzionare correttamente, necessita di interagire con altri servizi Hackystat come il principale server ovvero il SensorBase server, e Telemetry. Tali componenti forniscono accesso ai

dati solo ad utenti registrati sul SensorBase server. Inoltre se non si è l'amministratore (unico) del SensorBase server, è possibile accedere solo ai propri dati personali. Ciò condiziona quindi l'accesso anche ai dati forniti da LiSeD, che possono essere suddivisi in risorse accessibili da chiunque e in quelle accessibili solo da utenti registrati.

5.1.1.1 API PER UTENTI REGISTRATI

Solo gli utenti registrati sul server Hackystat d'interesse hanno i permessi necessari ad ottenere una risposta appropriata alle richieste seguenti.

Metodo	URI	Descrizione	Accessibilità
GET	{host}/ users	Elenco di URIs di tutti gli utenti registrati sul dato SensorBase server.	Amministratore
GET	{host}/ users / {user}	Descrizione del profile dell'utente specificato.	L'utente specificato deve coincidere con l'utente autenticato attualmente
GET	{host}/ projects	Elenco di URIs dei progetti SW.	Amministratore
GET	{host}/ projects / {user}	Elenco di URIs dei progetti in cui è coinvolto l'utente specificato.	L'utente specificato deve coincidere con l'utente autenticato attualmente
GET	{host}/ projects / {owner} / {project}	Descrizione del profilo del progetto SW specificato.	L'utente attualmente autenticato deve essere coinvolto nel progetto specificato.

GET	{host}/ issues	Elenco di URIs di issue.	Amministratore.
GET	{host}/ issues / {id}	Descrizione del profilo dell'issue specificato.	L'issue specificato dev'essere stato collezionato da un sensore associato all'utente autenticato.
GET	{host}/ users /sparql?query={query}	Risultati ricerca di URIs di utenti il cui profilo soddisfa la query specificata (codificata in UTF-8).	La ricerca avviene sui dati provenienti da tutti i server della propria Rete Hackstat, mentre localmente, a meno di essere autenticati come amministratore, si considerano solo il profilo dell'utente autenticato.
GET	{host}/ projects /sparql?query={query}	Risultati ricerca di URIs di progetti i cui profili soddisfano la query specificata (codificata in UTF) .	La ricerca avviene sui dati provenienti da tutti i server della propria Rete Hackstat, mentre localmente, a meno di essere autenticati come amministratore, si considerano solo i profili dei progetti in cui l'utente autenticato è

			coinvolto.
GET	{host}/ issues /sparql?query={query}	Risultati ricerca di URIs di issue i cui profili soddisfano la query specificata (codificata in UTF) .	La ricerca avviene sui dati provenienti da tutti i server della propria Rete Hackstat, mentre localmente, a meno di essere autenticati come amministratore, si considerano solo i profili degli issue collezionati dai sensori associati all'utente autenticato.
GET	{host}/ command / {user} / {command}	Descrizione del comando specificato, utilizzato dall'utente specificato. A seconda del livello di dettaglio dei dati dei sensori, ciò può includere argomenti passati al comando, risultato ottenuto, sistema operativo e calcolatore in	L'utente specificato deve coincidere con l'utente autenticato attualmente

		uso.	
GET	{host}/ devPhase /{user}/{project}/{phaseId}	Descrizione della fase di sviluppo SW specificata tramite il suo identificatore, associate al progetto specificato. A seconda della descrizione fornita dall'utente per tale fase, ciò può includere URIs delle fasi che precedono, degli utenti coinvolti e dei potenziali issue.	L'utente specificato deve coincidere con l'utente autenticato attualmente
POST	{host}/ network	Aggiunge un elenco di coppie nella forma (serverURI, adminPassw) alla locale lista di LiSeD servers partecipanti alla Rete Hackystat.	Amministratore
DELETE	{host}/ network /{uri}	Rimuove l'URI specificato dalla locale lista di LiSeD server	Amministratore

		partecipanti alla Rete Hackstat.	
DELETE	{host}/ cache	Cancella tutta la cache server-side.	Utente registrato sulla SensorBase
DELETE	{host}/ cache / {user}	Cancella tutta la cache server-side associata all'utente specificato.	L'utente specificato deve coincidere con l'utente autenticato attualmente
DELETE	{host}/ cache / {user} / {project}	Cancella tutta la cache server-side associata all'utente e al progetto specificati.	L'utente attualmente autenticato deve essere coinvolto nel progetto specificato.
DELETE	{host}/ cache /others	Cancella tutta la cache server-side non associata ad alcun utente.	Utente registrato sulla SensorBase

5.1.1.2 API PER UTENTI QUALSIASI

Qualsiasi utente anche se non registrato sul SensorBase server, può ottenere una risposta adeguata inoltrando una delle seguenti richieste.

Metodo	URI	Descrizione	Accessibilità
GET	{host}/ vocab	Descrizione del vocabolario Hackstat.	Anyone
GET	{host}/ vocab /void	Descrizione del locale dataset utilizzando void.	Anyone

GET	{host}/ programmingLanguage /{name}	Descrizione del linguaggio di programmazione specificato.	Anyone
GET	{host}/ operatingSystem /{name}	Descrizione del sistema operativo specificato.	Anyone
GET	{host}/ machine /{name}	Descrizione del calcolatore specificato	Anyone
GET	{host}/ tool /{name}	Descrizione del tool specificato.	Anyone
GET	{host}/ network	Elenco dei LiSeD server inclusi nella locale Rete Hackstat associate alle e-mail dei rispettivi amministratori.	Anyone
GET	{host}/ ping	"Ping representation" di questi servizio, ovvero la stringa "LinkedServiceData"	Anyone
GET	{host}/ ping ?user={user}&password={password}	Se l'utente e la password specificati sono credenziali valide, conferma, visualizzando lastringa "LinkedServiceData authenticated".	Anyone

5.1.1.3 DETTAGLI SULLE RISORSE

Vengono forniti di seguito dettagli riguardanti l'invocazione di ognuna delle risorse LiSeD e la relativa rappresentazione. Non sono previsti parametri addizionali nelle richieste inviate al LiSeD server, e l'unico caso in cui è accettata la ricezione di un payload è l'aggiunta (POST) di un elenco di server LiSeD da aggiungere alla locale lista di server partecipanti nella propria Rete Hackstat.

In aggiunta alle risorse approfondite di seguito, è accessibile dalla REST API anche un “Ping” URI, un modo semplice di verificare se un host risponde o meno alle richieste. Non è necessaria alcuna autenticazione. La risposta è la stringa “LinkedServiceData”. Se vengono forniti anche username e password allora la stringa di risposta è “LinkedServiceData authenticated” nel caso in cui le credenziali fornite siano valide. Tale autenticazione implica la verifica delle credenziali sul Sensorbase server sottostante.

Ciò che è stato comunemente definito un elenco di utenti o progetti o issue corrisponde semplicemente ad un RDF bag di URIs. Per esempio un elenco di utenti è simile alla seguente:

```
<http://localhost:9875/linkedservicedata/users/> http://www.w3.org/1999/02/22-rdf-syntax-ns#_4
http://localhost:9875/linkedservicedata/users/TestUserData@hackystat.org
<http://localhost:9875/linkedservicedata/users/> http://www.w3.org/1999/02/22-rdf-syntax-ns#_2
http://localhost:9875/linkedservicedata/users/TestUser2@hackystat.org
<http://localhost:9875/linkedservicedata/users/> http://www.w3.org/1999/02/22-rdf-syntax-ns#_3
http://localhost:9875/linkedservicedata/users/myriam.leggieri@gmail.com
<http://localhost:9875/linkedservicedata/users/> http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag ;
<http://localhost:9875/linkedservicedata/users/> http://www.w3.org/1999/02/22-rdf-syntax-ns#_1
http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it
```

Mentre un esempio di richiesta di ricerca utenti è la seguente:

```
http://localhost:9875/linkedservicedata/users/sparql?query=PREFIX+hacky%3A+%3Chttp%3A%2F%2Fdasha.ics.haw
aai.edu%3A9875%2Flinkedservicedata%2Fvocab%2F%3E%0APREFIX+xsd%3A+%3Chttp%3A%2F%2Fwww.w3.or
g%2F2001%2FXMLSchema%23%3E%0ASELECT+%3Furi%0A+WHERE+%7B%0A%3Furi+hacky%3A+communic
ationEffort+%3Fvalue0+.%0AFILTER+%28xsd%3Adouble%28%3Fvalue0%29+%3E+1.0%29%0A%7D
```

Nella query codificata usando lo schema UTF-8 è contenuto un solo filtro: communicationEffort>1.

Di seguito riporto la query non codificata:

```
PREFIX hacky: <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?uri
WHERE {
?uri hacky:communicationEffort ?value0 .
FILTER (xsd:double(?value0) > 1.0)
}
```

La risposta alle richieste di ricerca in generale, rispetta le direttive del W3C riguardanti lo “Sparql query result format”[29], ed è quindi in formato XML, come nell’esempio seguente:

```
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
<head>
```

```

<variable name="uri"/>
</head>
<results>
<result>
<binding name="uri">
<uri>http://localhost:9875/linkedservicedata/users/myrpanemon@yahoo.it</uri>
</binding>
</result>
</results>
</sparql>

```

5.1.1.3.1 RISORSE DI TIPO UTENTE

Il profile di un utente contiene tutte le asserzioni (soggetto, relazione, oggetto) aventi l'utente specificato come soggetto. Sono incluse quindi le seguenti relazioni.

Vocabolario	Nome Relazione	Descrizione oggetto
foaf	nick	Nickname
foaf	homepage	Homepage
foaf	weblog	Blog
foaf	firstName	Nome
foaf	mbox	E-mail
sioc	subscriber_of	Channel frequentati
dcTerm	isPartOf	Descrizione dataset
rdf	type	Classe a cui il soggetto appartiene
rdfs	seeAlso	Collegamento ad una risorsa potenzialmente correlabile
owl	sameAs	Collegamento ad una risorsa che a riferimento esattamente allo stesso soggetto.
hacky	communicationEffort	Quantità di impegno impiegato nel comunicare con gli altri.
Hacky	unitTestQuality	Rapporto tra Unit Test terminati con successo e Unit Test falliti +1.

hacky	commitFrequency	Frequenza di commit effettuati (media)
hacky	amountOfCodeIssues	Quantità di code issue generati (media)
hacky	buildQuality	Rapporto tra build terminati con successo e build falliti +1.
hacky	developmentTime	Tempo medio impiegato in attività di sviluppo SW.
hacky	karma	Il Karma si riferisce al principio universale secondo cui un'azione a fin di bene genera benefici nelle vite future. Misura globale sulla qualità dello sviluppatore.
hacky	privacyLevel	Livello di privacy

Le informazioni su cui si basano i calcoli sulla qualità dello sviluppatore, provengono dai dati dei sensori rielaborati dalla componente Telemetry, collezionati su tutti i progetti in cui l'utente è coinvolto. Tuttavia i dati considerati risalgono alla data di creazione di ogni progetto solo nel caso in cui sia trascorso meno di un anno da tale data; altrimenti si considerano solo i dati collezionati dall'inizio dell'anno corrente, per motivi di complessità in tempo.

In particolare l'impegno nella comunicazione è dato dalla quantità totale di messaggi inviati sui canali IRC monitorati da SiocLog [35], mentre il valore Karma consiste semplicemente nella somma dei valori ottenuti da tutte le altre misure di qualità. E' evidente che tali misure di qualità utilizzate per stimare l'affidabilità di uno sviluppatore da determinati punti di vista, siano approssimative e provvisorie. Misure più precise ed efficaci sono necessarie, ma individuarle avrebbe richiesto un approfondimento ulteriore, non contemplato nel presente progetto.

Il livello di privacy è un campo aggiunto per supportare un auspicato sviluppo futuro di hackystat, verso una gestione più flessibile della privacy degli utenti, indispensabile ad un pieno sviluppo dei Linked Data, contribuendo ad accrescere la quantità di dataste online invece che limitarsi a sfruttare i dataset pubblicati da altri. Si auspica che in futuro la SensorBase permetta agli utenti di decidere se preferiscono che i loro dati siano visibili:

- Solo a se stessi e all'amministratore del SensorBase server
- Solo agli utenti registrati sul SensorBase server
- A chiunque

Con la possibilità di impostare uno di tali livelli di privacy per ognuno delle tre principali tipologie di dati, ovvero: profilo utente, progetto SW e issue.

Un esempio di risposta supponendo che si sia richiesto il linguaggio di rappresentazione RDF N-triple) è la seguente:

```
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://xmlns.com/foaf/0.1/nick> "iammyr" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://www.w3.org/2000/01/rdf-
schema#seeAlso> "http://irc.sioc-project.org/users/iammyr#user" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://www.w3.org/2000/01/rdf-
schema#seeAlso> "http://rdfohloh.wikier.org/user/52334" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/communicationEffort> "96" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/commitFrequency> "0" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://xmlns.com/foaf/0.1/weblog>
"http://myr.netsons.org/wordpress" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://xmlns.com/foaf/0.1/homepage>
"http://myr.altervista.org" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/amountOfCodeIssues> "2.5" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://rdfs.org/sioc/ns#subscriber_of>
<http://irc.sioc-project.org/foaf#channel> .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://dublincore.org/documents/dcmi-
terms/#terms-isPartOf> "http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/void" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://xmlns.com/foaf/0.1/mbox>
"myrpandemon@yahoo.it" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://www.w3.org/2002/07/owl#sameAs>
"http://rdfohloh.wikier.org/user/52334" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://xmlns.com/foaf/0.1/firstName>
"Myriam" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/karma> "175.83333333333331" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://www.w3.org/2002/07/owl#sameAs>
"http://myr.netsons.org/foaf.rdf#iammyr" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/buildQuality> "1" .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://rdfs.org/sioc/ns#subscriber_of>
<http://irc.sioc-project.org/sioc#channel> .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it> <http://www.w3.org/1999/02/22-rdf-syntax-
ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/developmentTime> "76.33333333333333" .
```

5.1.1.3.2 RISORSE DI TIPO PROGETTO

Il profilo di un progetto contiene tutte le asserzioni (soggetto, relazione, oggetto) aventi il progetto specificato come soggetto. Sono incluse quindi le seguenti relazioni.

Vocabulary	Relationship Name	Object Description
doap	name	Nome
doap	maintainer	Project Mantainer
doap	created	Data in cui il progetto è stato creato
doap	repository	URI del repository
doap	anon-root	URI del repository che consente accesso anonimo
doap	category	Categoria (Tag) a cui il progetto appartiene
doap	browse	URI dell'interfaccia web del repository
hacky	includesPhase	Fase di sviluppo SW prevista nel progetto
hacky	usesTool	Tool utilizzato
hacky	modified	Data ultima modifica
hacky	ended	Data in cui il progetto è stato chiuso
hacky	projectOwner	Proprietario del progetto
hacky	amountOfCodeIssues	Quantità di code issue generati (media)
hacky	buildQuality	Rapporto tra build terminati con successo e build falliti +1.
hacky	unitTestQuality	Rapporto tra Unit Test terminati con successo e Unit Test falliti +1.
hacky	coverageQuality	Percentuale di copertura raggiunta con livello di granularità pari a quello di classe
hacky	commitFrequency	Frequenza di commit effettuati dai membri (media)
hacky	developmentTime	Media tra le quantità di tempo impiegato dai membri del

		progetto in attività di sviluppo sul progetto stesso.
hacky	privacyLevel	Livello di privacy
RDFS	seeAlso	Collegamento ad una risorsa potenzialmente correlabile
dcTerm	isPartOf	Descrizione dataset

Un esempio di risposta supponendo che si sia richiesto il linguaggio di rappresentazione RDF N-triple) è la seguente:

```
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#name> "LinkedServiceData_Project" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#maintainer> "http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#created> "2009-08-22T20:24:11.934-10:00" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/includesPhase>
"http://localhost:9875/linkedservicedata/devPhase/myrpandemon@yahoo.it/LinkedServiceData_Project/Documentation
" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/61048" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/60164" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/87453" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/386892" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#repository> "" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#anon-root> "" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://usefulinc.com/ns/doap#category> "RDF" .
<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/includesPhase>
"http://localhost:9875/linkedservicedata/devPhase/myrpandemon@yahoo.it/LinkedServiceData_Project/Analysis" .
```

<http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dublincore.org/documents/dcmi-terms/#terms-isPartOf>
 "http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/void" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://usefulinc.com/ns/doap#browse> "" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/61150" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/projectOwner>
 "http://localhost:9875/linkedservicedata/users/myrpandemon@yahoo.it" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/modified> "2009-08-22T20:31:35.464-10:00" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/ended> "2010-08-22T00:00:00.000-10:00" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/usesTool>
 "http://localhost:9875/linkedservicedata/tool/LinkedData" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://usefulinc.com/ns/doap#Project> .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/privacyLevel> "" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/399312" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/391948" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/usesTool>
 "http://localhost:9875/linkedservicedata/tool/RDF" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2002/07/owl#sameAs> "http://rdfohloh.wikier.org/project/377043" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://usefulinc.com/ns/doap#category> "LinkedData" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/75592" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/95823" .
 <http://localhost:9875/linkedservicedata/projects/myrpandemon@yahoo.it/LinkedServiceData_Project>
 <http://www.w3.org/2000/01/rdf-schema#seeAlso> "http://rdfohloh.wikier.org/project/22189" .

5.1.1.3.3 RISORSE DI TIPO ISSUE

Il profilo di un issue contiene tutte le asserzioni (soggetto, relazione, oggetto) aventi l'issue specificato come soggetto. Sono incluse quindi le seguenti relazioni.

Vocabolario	Nome Relazione	Descrizione Oggetto
baetle	priority	Priorità
baetle	target_milestone	Target milestone
baetle	assigned_to	Utente a cui è stata assegnata la risoluzione dell'issue
baetle	name	Nome
baetle	status	Status
baetle	created	Data di creazione
baetle	duplicate	URI di un issue segnata come duplicata della corrente
commonTag	tagged	Tag assegnato
hacky	closedTime	Data di chiusura
hacky	modifiedTime	Data ultima modifica
owl	sameAs	Collegamento ad una risorsa che fa riferimento esattamente allo stesso soggetto
dcTerm	isPartOf	Descrizione dataset

Un esempio di risposta supponendo che si sia richiesto il linguaggio di rappresentazione RDF N-triple) è la seguente:

```
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/baetle/#priority> "critical" .
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/baetle/#target_milestone> "milestone4" .
<http://localhost:9875/linkedservicedata/issues/101> <http://commontag.org/ns#tagged> "browser" .
<http://localhost:9875/linkedservicedata/issues/101>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/closedTime> "2009-06-20T00:00:00.000-10:00" .
<http://localhost:9875/linkedservicedata/issues/101> <http://www.w3.org/2002/07/owl#sameAs>
"http://localhost:9875/linkedservicedata/issues/456" .
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/baetle/#assigned_to>
"http://localhost:9875/linkedservicedata/users/john@yahoo.it" .
```

```

<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/foaf/#name> "101" .
<http://localhost:9875/linkedservicedata/issues/101> <http://dublincore.org/documents/dcmi-terms/#terms-isPartOf>
"http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/void" .
<http://localhost:9875/linkedservicedata/issues/101> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://xmlns.com/foaf/#Defect> .
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/foaf/#status> "fixed" .
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/foaf/#created> "2006-10-12T00:00:00.000-
10:00" .
<http://localhost:9875/linkedservicedata/issues/101> <http://commontag.org/ns#tagged> "gui" .
<http://localhost:9875/linkedservicedata/issues/101>
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/modifiedTime> "2009-05-07T00:00:00.000-10:00" .
<http://localhost:9875/linkedservicedata/issues/101> <http://commontag.org/ns#tagged> "mysql" .
<http://localhost:9875/linkedservicedata/issues/101> <http://xmlns.com/foaf/#duplicate>
"http://localhost:9875/linkedservicedata/issues/112" .

```

5.1.1.3.4 RISORSE DI TIPO NETWORK

Una risorsa di tipo Network consiste in un elenco di coppie nella forma <serverURI, adminPassword>. Se un client lo richiede, tale elenco viene fornito in risposta, privato dei valori relativi alle password degli amministratori.

Inoltre un client può inviare un elenco simile, contenente cioè lo stesso tipo di coppie, richiedendo che sia aggiunto all'elenco locale. In tal caso deve inviarlo come payload nella forma:

```
uri1,email1__passw1 uri2,email2__passw2 ... urin,emailn__passwn
```

Tale forma è stata preferita al formato XML in quanto più facilmente parserizzabile.

Se un cliente desidera cancellare una coppia dall'elenco di server inclusi nella locale Rete Hackystat, è sufficiente che specifichi l'URI del server da rimuovere, codificato secondo lo schema UTF-8, inviando dunque una richiesta simile alla seguente:

DELETE

```
http://dasha.ics.hawaii.edu:9875/linkedservicedata/network/http%3A%2F%2Flocalhost%3A9875%2Flinkedservicedata%2F
```

5.1.1.3.5 RISORSE AD ACCESSO LIBERO

Le risorse accessibili anche da utenti non registrati sul SensorBase server, consistono in descrizioni di linguaggi di programmazione, sistemi operativi, tool, calcolatori. Tali descrizioni includono la seguente relazione.

Vocabolario	Nome Relazione	Descrizione Oggetto
-------------	----------------	---------------------

rdfs	seeAlso	Collegamento ad una risorsa potenzialmente correlabile
------	---------	--

5.1.1.3.6 RISORSA DI TIPO DATASET HACKYSTAT

Il “*Vocabulary of Interlinked Datasets*” [16] è stato creato ed è in fase di sviluppo al fine di superare il ben noto problema del “*Web of Data Discovery*”, ovvero le difficoltà incontrate nel reperire tutti dataset focalizzati su un certo topic di interesse, tramite Semantic Search Engine. La causa principale delle difficoltà risiede nell’utilizzo in tali dataset di differenti vocabolari per descrivere le stesse relazioni, gli stessi concetti. In particolare non esiste un vocabolario universale mirato alla descrizione del dataset stesso, che comunichi quindi al semantic search engine, il dominio su cui è focalizzato. VOID mira a colmare questa lacuna, fornendo un vocabolario per la descrizione dei dataset in cui è possibile specificare il dominio di riferimento, eventuali SPARQL endpoint e i regex pattern utilizzati.

Un ulteriore problema però è costituito dall’identificatore da assegnare al nome del dominio di riferimento. Si è scelto dunque di riporre fiducia nel processo di diffusione di Umbel utilizzando, per identificare i concetti di dominio, gli URI di più di 200.000 subject concept in esso contenuti. L’Hackystat dataset è stato descritto utilizzando void e Umbel, e in particolare i concetti SoftwareDeveloper, SoftwareProject e SoftwareIssue definiti in Umbel. La risposta ad una richiesta di descrizione del dataset tramite REST API, specificando N3 come linguaggio di serializzazione RDF:

```
<http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://rdfs.org/ns/void#Dataset> ;
  <http://purl.org/dc/elements/1.1/creator>
    "http://foafbuilder.qdos.com/people/myriamleggieri.wordpress.com/foaf.rdf#iammyr" ;
  <http://purl.org/dc/elements/1.1/description>
    "Data about users, projects and issues stored in the Hackystat SensorBase and/or further handled by the
Hackystat Telemetry service published as Linked Data." ;
  <http://purl.org/dc/elements/1.1/publisher>
    [ <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
```



```

    <http://xmlns.com/foaf/0.1/Organization> ;
    <http://www.w3.org/2000/01/rdf-schema#label>
        "Hackystat - Collaborative Software Development Laboratory - Department of Information and
        Computer Sciences - University of Hawaii at Manoa" ;
    <http://xmlns.com/foaf/0.1/homepage>
        "http://csdl.ics.hawaii.edu/research/hackystat"
    ] ;
    <http://purl.org/dc/elements/1.1/subject>
        "http://umbel.org/umbel/sc/SoftwareProject", "http://umbel.org/umbel/sc/SoftwareDeveloper",
        "http://umbel.org/umbel/sc/SoftwareIssue" ;
    <http://purl.org/dc/elements/1.1/title>
        "Hackystat_LinkedServiceData" ;
    <http://purl.org/dc/terms/accessRights>
        "http://www.gnu.org/copyleft/fdl.html" ;
    <http://rdfs.org/ns/void#sparqlEndpoint>
        "http://dasha.ics.hawaii.edu:9875/linkedservicedata/users/sparql?query=",
        "http://dasha.ics.hawaii.edu:9875/linkedservicedata/projects/sparql?query=",
        "http://dasha.ics.hawaii.edu:9875/linkedservicedata/issues/sparql?query=" ;
    <http://rdfs.org/ns/void#uriRegexPattern>
        ".*projects/sparql?query=.*", ".*users.*", ".*issues/sparql?query=.*", ".*issues.*", ".*projects.*",
        ".*users/sparql?query=.*" ;
    <http://rdfs.org/ns/void#vocabulary>
        "http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/",
        "http://www.ifi.uzh.ch/ddis/evoont/2008/11/vom#", "http://commontag.org/ns#", "http://xmlns.com/foaf/0.1/",
        "http://purl.org/dc/elements/1.1/", "http://www.itee.uq.edu.au/~dwood/ontologies/sec.owl#",
        "http://annotation.semanticweb.org/iswc/iswc.daml#", "http://rdfs.org/ns/void#", "http://picoforge.int-
        evry.fr/projects/svn/helios_wp3/2009/07/helios_bt.owl#", "http://xmlns.com/baetle/#", "http://rdfs.org/sioc/ns#",
        "http://umbel.org/umbel/sc/", "http://usefulinc.com/ns/doap#", "http://www.daml.org/services/owl-s/1.1/Process.owl#",
        "http://purl.org/dc/terms/", "http://www.w3.org/2002/07/owl#" .

```

5.1.1.3.7 RISORSE DI TIPO CACHE

Il servizio LinkedSensorData supporta un sistema di caching delle istanze LinkedSensorData denominato “Front-side cache”. Il LiSeD caching può migliorare significativamente le prestazioni, riducendo la quantità di chiamate ai servizi SensorBase e Telemetry, e la conseguente computazione (creazione di modelli RDF) su tali istanze. Tuttavia quando i dati relativi ad un’istanza cambiano, la corrispondente istanza memorizzata in cache potrebbe non essere più valida. In tal caso l’utente necessita di cancellare le istanze di interesse dalla cache. Non è possibile specificare esattamente l’istanza da cancellare, bensì è possibile specificare una vista sulla cache, in cui essa rientra. Sono infatti individuate tre diverse viste e le relative seguenti richieste al servizio LiSeD.

Cancellamento dati nella vista comprendente tutte le istanze associate all'utente autenticato:

DELETE http://dasha.ics.hawaii.edu:9875/linkedservicedata/cache/

Cancellamento dati nella vista comprendente tutte le istanze di uno specificato progetto avente uno specificato possessore, a condizione che l'utente autenticato sia coinvolto in tale progetto (altrimenti viene visualizzato un messaggio di errore):

DELETE http://dasha.ics.hawaii.edu:9875/linkedservicedata/cache/johnson@hawaii.edu/TestProject

Cancellamento dati nella vista comprendente tutte le istanze associate ad uno specificato utente, a condizione che l'utente autenticato coincida con l'utente specificato a meno che l'utente autenticato non sia l'amministratore:

DELETE http://dasha.ics.hawaii.edu:9875/linkedservicedata/cache/johnson@hawaii.edu

Cancellamento dati nella vista comprendente tutte le istanze non associate ad alcun utente, le quali coincidono con le risorse qui definite come prive di restrizioni sull'accesso:

DELETE http://dasha.ics.hawaii.edu:9875/linkedservicedata/others

L'abilitazione del sistema di caching è opzionale. Nel caso in cui esso sia disabilitato, le richieste suddette al servizio LiSeD non producono alcun effetto.

5.1.2 UTILIZZO DI UN CLIENT JAVA

La distribuzione LinkedSensorData include una classe denominata LinkedServiceDataClient che può essere utilizzata in applicazioni Java esterne per ottenere risorse LiSeD. La JavaDoc associata a tale classe ne fornisce tutti i dettagli su tale API (vedi Appendice A).

Un Java code sample che utilizza tale API per effettuare una ricerca sulle risorse di tipo Progetto è il seguente:

```
LinkedServiceDataClient lisedClient1 = new LinkedServiceDataClient(admin, passw, lang, media, true, true);
```

```
    lisedClient1.authenticate();
    lisedClient1.clearServerCache();
    lisedClient1.clearLocalCache();
    ProjectQuery pq = null;
    pq = new ProjectQuery();
    pq.projectname = projectName;
    pq.owner = user;
    pq.start = start.toXMLFormat();
    pq.end = end.toXMLFormat();
    pq.progrlangs = progrlangs;
    pq.osystems = osystems;
    pq.bugdatabases = bugdatabases;
```

```

pq.mirrors = downloadmirrors;
pq.repwebinterface = repwebinterface;
pq.wikis = wikis;
pq.tags = tags;
pq.tools = tools;
pq.usersRoles = usersRoles;
String ret = lisedClient1.searchForProjects(pq);
assertTrue(ret != null);
System.out.println(ret);

```

Per utilizzarlo è necessario incollarlo in una propria classe Java e inizializzare con un qualunque valore si preferisca le variabili in esso utilizzate. E' anche necessario includere linkedservicedata.jar nel proprio CLASSPATH e assicurarsi che i server SensorBase, DailyProjectData, Telemetry e LinkedServiceData siano avviati e in esecuzione. Un esempio di risultato ottenuto è il seguente:

```

<sparql xmlns="http://www.w3.org/2005/sparql-results#">
<head>
<variable name="uri"/>
</head>
<results>
<result>
<binding name="uri">
<uri>http://localhost:9875/linkedservicedata/projects/holatest@hackystat.org/Holatest3</uri>
</binding>
</result>
</results>
</sparql>

```

5.1.3 UTILIZZO DELLA GUI

E' possibile utilizzare anche un'interfaccia grafica (Graphical User Interface, GUI) standalone per interagire con il LiSeD server. L'URI del LiSeD server dev'essere specificato in locale nel file linkedservicedata.properties. Sia che tale file esista già a causa dell'avvi del LiSeD server avvenuto sulla stessa macchina su cui si desidera utilizzare la GUI, e sia che invece tale file non esista e necessiti di essere creato, esso deve avere il seguente path:

```
~/hackystat/linkedservicedata/linkedservicedata.properties
```

E deve contenere la specifica dell'host su cui si trova il LiSeD server cui si vuol fare riferimento, come descritto di seguito:

Property	Default	Description
linkedservicedata.hostname	localhost	The host name, i.e. http://{hostname}:{port}/{context.root}/
linkedservicedata.context.root	linkedservicedata	The context root, i.e. http://{hostname}:{port}/{context.root}/
linkedservicedata.port	9875	The port, i.e. http://{hostname}:{port}/{context.root}/

Esempio:

linkedservicedata.hostname=dasha.ics.hawaii.edu

linkedservicedata.context.root=linkedservicedata

linkedservicedata.port=9875

L'interfaccia grafica si connette al LiSeD server utilizzando le informazioni inserite nel file .properties. La GUI presenta in apertura tutti i tab vuoti in quanto non ci si è ancora autenticati. E' possibile autenticarsi tramite l'interfaccia di login nel tab 'Authentication' (Fig. 5.3)



Figura 5.3 – Autenticazione tramite interfaccia grafica

Dopo essersi autenticati tutti i servizi LiSeD diventano accessibili. In qualsiasi tab si scelga di visualizzare sono sempre fornite le seguenti funzionalità:

- Pulire la server-side cache (tramite bottone "Clear cache")

- Visualizzare la rappresentazione RDF di tutte le informazioni contenute attualmente nella finestra di interazione (tramite bottone “Triplify All”)
- Scegliere il linguaggio di serializzazione RDF preferito (tramite menù a tendina posizionato nell’angolo in alto a destra)
- Pulire la client-side cache (tramite bottone “Refresh”)

Per specificare il linguaggio di serializzazione RDF che si desidera venga utilizzato nelle rappresentazione delle risorse, dopo aver selezionato il linguaggio preferito nel menù a tendina in alto a destra, è necessario pulire la client-side cache cliccando sul bottone “Refresh”.

Accedere alle tab denominate Users, Projects o Issues causa la visualizzazione di una lista completa di rispettivamente, tutti i profili utente, progetto o issue accessibili in base ai permessi assegnati all’utente autenticato. Tali elenchi completi vengono caricati per default appena si clicca su un tab.

5.1.3.1 PANNELLO UTENTI

Nel pannello Utenti cui si accede cliccando sul tab “Users” sono elencati uno o più profili utente a seconda dei permessi assegnati all’utente correntemente autenticato. In attesa di futuri sviluppi riguardo più flessibili livelli di privacy, solo l’amministratore può visualizzare l’elenco di tutti gli utenti Hackstat registrati sul SensorBase server di riferimento (Fig. 5.4).

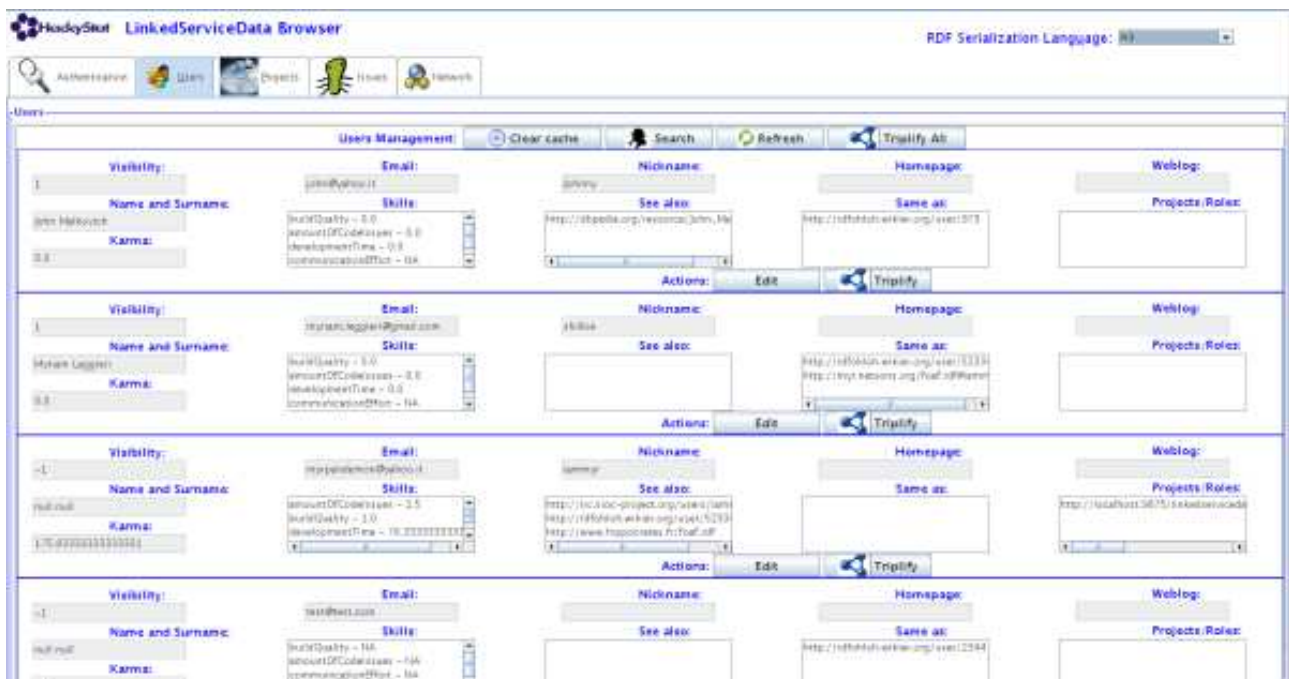


Figura 5.4 - Visualizzazione del Pannello Utenti in caso di utente identificato come amministratore.

Di conseguenza se non si è stati identificati come amministratori verrà visualizzato solo il proprio personale profilo utente (Fig. 5.5).



Figura 5.5 Visualizzazione del Pannello Utenti in caso di utente non identificato come amministratore.

Come è possibile vedere in Figura 5.4 e 5.5, ad ogni profilo utente è associato un bottone “Edit” per modificare i dettagli del profilo. Si è preferito evitare che l’amministratore potesse, oltre che visualizzare tutti i profili degli utenti, anche modificarli. Di conseguenza qualunque utente autenticato che tenti di modificare un profilo non suo, deve necessariamente inserire la password associata a quel dato profilo. In tal modo non avvengono modifiche ai profili senza che il proprietario di ognuno non ne sia al corrente, in quanto egli deve perlomeno aver fornito la sua password (segreta anche per l’amministratore) a qualcun altro incaricato eventualmente di modificare il profilo).

In fase di modifica l’interfaccia visualizzata cambia unicamente in quanto alcuni campi testuali diventano modificabili (Fig. 5.6), mentre sono altrimenti tutti disabilitati (Fig. 5.7), ed un bottone “Save” per salvare le modifiche effettuate, sostituisce il precedente bottone “Edit”.



Figura 5.6 – I campi testuali del profilo utente che si è scelto di editare, diventano modificabili.

Visibility: []

Email: [myrpandemon@yahoo.it]

Nickname: [myrpandemon]

Homepage: []

Weblog: []

Name and Surname: [myrpandemon]

Skills: []

Karma: [175.83333333333333]

Projects/Role: []

Actions: [Edit] [Triplify]

Figura 5.7 – Profilo utente prima che ne venga richiesta (e accettata) la modifica da parte dell'utente. Ad ogni profilo utente è associato anche un bottone “Triplify” per visualizzare la rappresentazione RDF del corrispondente profilo utente, in una nuova finestra a dimensioni ridotte (Fig. 5.8), utilizzando il linguaggio di serializzazione RDF che è stato impostato come preferito (utilizzando il menù a tendina in alto a destra come precisato in precedenza).

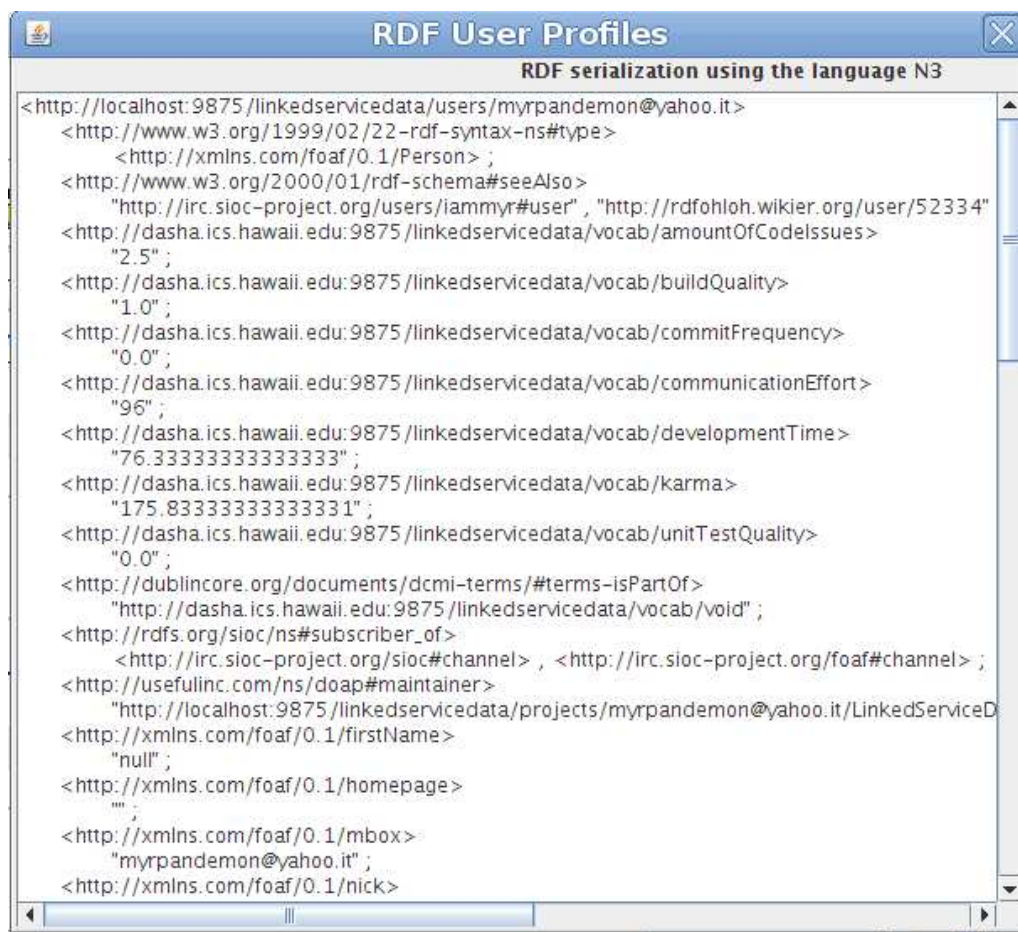
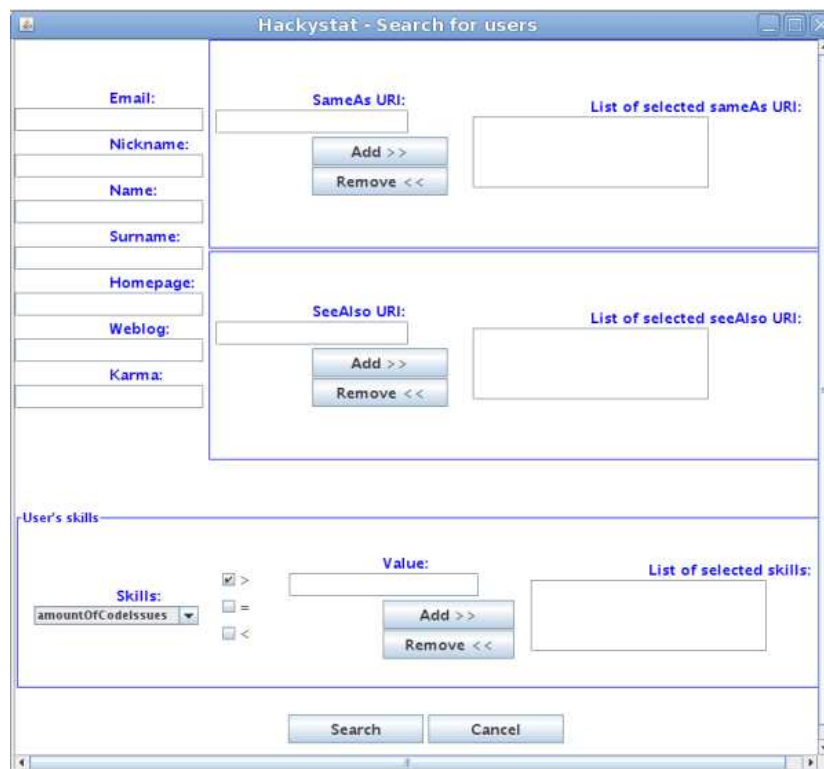


Figura 5.8 – Rappresentazione RDF del profilo di un utente in cui è stato utilizzato il linguaggio di serializzazione N3.

Il bottone “Triplify All” posizionato all’inizio dell’elenco profili, ha invece l’effetto di visualizzare in una finestra nuova a dimensioni ridotte, la rappresentazione RDF dell’elenco di profili stesso. Il bottone “Clear cache” elimina dalla server-side cache il contenuto della vista-cache che include le istanze LiSeD associate all’utente autenticato; mentre il bottone “Search” apre un wizard per la ricerca sugli utenti, in una nuova finestra (Fig. 5.9).



The image shows a window titled "Hackystat - Search for users". It contains several input fields for user profile information: Email, Nickname, Name, Surname, Homepage, Weblog, and Karma. There are two sections for adding URIs: "SameAs URI" and "SeeAlso URI", each with an "Add >>" button, a "Remove <<" button, and a "List of selected" area. Below these is a section for "User's skills" with a "Skills:" dropdown menu (currently showing "amountOfCodeIssues"), a "Value:" input field, and "Add >>" and "Remove <<" buttons. At the bottom are "Search" and "Cancel" buttons.

Figura 5.9 – Query Wizard per effettuare ricerche sugli Utenti.

La ricerca è effettuabile su uno qualsiasi dei campi che compongono il profilo dell'utente. In particolare notiamo la ricerca in base alle capacità dello sviluppatore che consente di individuare con affidabilità colleghi di lavoro a cui rivolgersi per chiedere aiuto o suggerimento su particolari questioni. I risultati di ricerca vengono visualizzati in una nuova finestra (Fig. 5.10) e consistono sempre in un elenco di URI di risorse che soddisfano le caratteristiche specificate nel wizard.



The image shows a window titled "Search Results". It contains a list of search results under the heading "Search Results:". The results are a list of URIs found, all starting with "http://localhost:9875/linkedservicedata/users/". The URIs include usernames like "TestUser@hackystat.org", "TestUser3@hackystat.org", "myrpandemon@yahoo.it", "test@test.com", "TestUser2@hackystat.org", "myriam.leggieri@gmail.com", "TestTelPing@hackystat.org", "TestUserData@hackystat.org", and "john@yahoo.it".

Figura 5.10 – Risultati ricerca sui profili utente

La ricerca, come precedentemente specificato, viene propagata alla Rete Hackystat.

5.1.3.2 PANNELLO PROGETTI

Nel pannello Progetti cui si accede cliccando sul tab “Projects” sono elencati uno o più profili progetto a seconda dei permessi assegnati all’utente correntemente autenticato. In attesa di futuri sviluppi riguardo più flessibili livelli di privacy, solo l’amministratore può visualizzare l’elenco di tutti i progetti Hackystat registrati sul SensorBase server di riferimento. Se invece non si è stati identificati come amministratori verranno visualizzati solo i profili dei progetti in cui l’utente autenticato è coinvolto (Fig. 5.11).

The screenshot displays the 'Project Management' interface. At the top, there's a 'Projects Management' section with buttons for 'Clear cache', 'Search', 'Refresh', and 'Trigify All'. Below this, the form is organized into several sections:

- Visibility:** Includes fields for 'Project Name', 'Owner', 'Anonymous repository', 'Repository', and 'Repository Web interface'.
- Start date:** A date field showing '18-08-2012 09:41:00'.
- Wiki:** A text area for project documentation.
- Programming Languages:** A list box for selecting programming languages.
- Mirrors:** A list box for selecting mirrors.
- Last modification date:** A date field showing '18-08-2012 09:41:00'.
- Bug Databases:** A list box for selecting bug databases.
- Operating systems:** A list box for selecting operating systems.
- Description:** A text area for project description.
- End date:** A date field showing '18-08-2012 09:41:00'.
- Same as:** A list box for selecting related projects.
- See also:** A list box for selecting related projects.
- Quality measures:** A list box for selecting quality measures.
- Member Role list:** A list box for selecting member roles.
- Development Phase:** A list box for selecting development phases.
- Tag:** A text field for tagging the project.
- Tools:** A list box for selecting tools.

At the bottom, there's an 'Actions' section with 'Edit' and 'Trigify' buttons.

Figura 5.11 - Visualizzazione del Pannello Progetti in caso di utente non identificato come amministratore.

Come è possibile vedere in Figura 5.11, ad ogni profilo progetto è associato un bottone “Edit” per modificare i dettagli del profilo. In fase di modifica l’interfaccia visualizzata cambia unicamente in quanto alcuni campi testuali diventano modificabili (Fig. 5.12), mentre sono altrimenti tutti disabilitati, ed un bottone “Save” per salvare le modifiche effettuate, sostituisce il precedente bottone “Edit”. In particolare alcuni dettagli di profilo sono costituiti da liste i cui elementi è possibile aggiungere sequenzialmente.

Figura 5.12 – I campi testuali e le liste del profilo progetto che si è scelto di editare, diventano modificabili.

Ad ogni profilo progetto equivalente a ciò che accade con i profili utente, sono associati un bottone “Triplify”, “Triplify All”, “Clear cache” e “Search” che rispettivamente si riferiscono alla rappresentazione RDF di un profilo progetto e dell’elenco di tutti i profili, alla cancellazione della cache server-side e all’apertura di un wizard per la ricerca sui progetti, in una nuova finestra (Fig. 5.13).

Figura 5.13 – Query Wizard per effettuare ricerche sui Progetti.

La ricerca è effettuabile su uno qualsiasi dei campi che compongono il profilo del progetto. In particolare notiamo la ricerca in base ai tool utilizzati e al dominio di applicazione, che consente di individuare progetti candidabili ad esempi pratici potenziali chiarificatori di dubbi sull’utilizzo di

tool e scelte progettuali legate al dominio. I risultati di ricerca vengono visualizzati in una nuova finestra e consistono sempre, equivalentemente ai risultati delle ricerche sugli utenti, in un elenco di URI di risorse che soddisfano le caratteristiche specificate nel wizard.

5.1.3.3 PANNELLO ISSUE

Nel pannello Issue cui si accede cliccando sul tab “Issues” sono elencati uno o più profili issue a seconda dei permessi assegnati all’utente correntemente autenticato. Gli issue vengono rilevati dai sensori a loro volta associati agli utenti. In attesa di futuri sviluppi riguardo più flessibili livelli di privacy, solo l’amministratore può visualizzare l’elenco di issue rilevati dai sensori di tutti gli utenti Hackystat (Fig. 5.14). Se invece non si è stati identificati come amministratori verranno visualizzati solo i profili degli issue collezionati dai sensori associati all’utente correntemente autenticato.

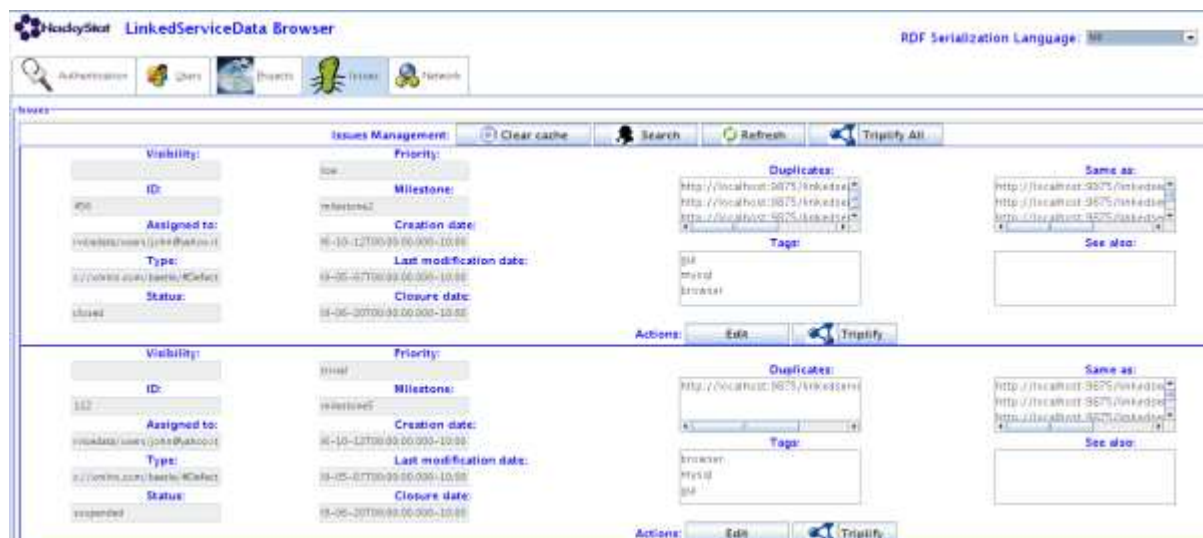


Figura 5.14 - Visualizzazione del Pannello Issue in caso di utente identificato come amministratore.

Il rilevamento di una lista completa di profili di issue è l’attività più costosa in termini di tempo tra quelle svolte. Tuttavia è in fase di sviluppo una versione della SensorBase caratterizzata da una migliore gestione dei dati dei sensori riguardanti gli issue. Tale versione è stata sviluppata durante lo stesso Google Summer of Code 2009 dallo studente PhD Shaoxuan.

Come è possibile vedere in Figura 5.14, ad ogni profilo progetto è associato un bottone “Edit” per modificare i dettagli del profilo. In fase di modifica l’interfaccia visualizzata cambia unicamente in quanto alcuni campi testuali diventano modificabili, mentre sono altrimenti tutti disabilitati, ed un bottone “Save” per salvare le modifiche effettuate, sostituisce il precedente bottone “Edit”. Ad ogni profilo issue equivalentemente a ciò che accade con i profili utente, sono associati un bottone “Triplify”, “Triplify All”, “Clear cache” e “Search” che rispettivamente si riferiscono alla

rappresentazione RDF di un profilo issue e dell'elenco di tutti i profili, alla cancellazione della cache server-side e all'apertura di un wizard per la ricerca sugli issue, in una nuova finestra (Fig. 5.15).

Figura 5.15 – Query Wizard per effettuare ricerche sugli Issue.

La ricerca è effettuabile su uno qualsiasi dei campi che compongono il profilo dell'issue. In particolare notiamo la ricerca in base ai tag e all'URI dell'utente cui l'issue è stato assegnato o origine della segnalazione. Ciò costituisce un'interconnessione tra risorse di tipo issue e di tipo utente. Inoltre l'associazione dell'URI di una milestone all'issue costituisce un'ulteriore connessione tra gli issue e i progetti. I risultati di ricerca vengono visualizzati in una nuova finestra e consistono sempre, equivalentemente ai risultati delle ricerche sugli utenti e sui progetti, in un elenco di URI di risorse che soddisfano le caratteristiche specificate nel wizard.

5.1.3.4 PANNELLO NETWORK

Nel pannello Network cui si accede cliccando sul tab “Network” sono elencati gli URI dei server host e le e-mail dei rispettivi amministratori, i quali hanno accettato di condividere le proprie informazioni con gli utenti del server LiSeD cui fa riferimento l'interfaccia grafica (Fig. 5.16).



Figura 5.16 – Pannello di gestione della Rete Hackstat

In tale pannello è possibile aggiungere nuove triple del tipo URI del server host, e-mail dell'amministratore, password dell'amministratore (Fig. 5.17).



Figura 5.17 – Aggiunta o rimozione di server LiSeD che partecipano al programma di condivisione dati con il server LiSed associato alla GUI.

L'aggiunta e la rimozione sono consentiti solo agli utenti identificati come amministratori; in caso si tenti di effettuare tali operazioni senza essere amministratori, viene visualizzato il messaggio di errore in Figura 5.18.



Figura 5.18 – Messaggio di errore visualizzato in caso di tentativi aggiunta/rimozione server nella Rete Hackystat da parte di utenti non amministratori.

5.2 LINEE GUIDA SEGUITE NELLA PUBBLICAZIONE DEI LINKED DATA

Nella pubblicazione dei Linked Sensor Data sono state seguite le linee guida consigliate da C. Bizer, R. Cyganiak e T. Heath in [38].

1. Le risorse pubblicate sono tutte informative tranne quelle riguardanti gli sviluppatori che, in quanto corrispondenti ad entità concrete nel mondo reale, sono risorsa non informativa. Nel contesto dei Linked Data tale distinzione è abbastanza importante e differenzia dal Web of Document in cui le risorse disponibili si limitano solo a quelle informative.
2. Le risorse sono tutte identificate da URI http, seguendo a grandi linee lo schema {LiSeD_host}/{entità}/{ID} ma per maggiori dettagli è possibile fare riferimento alla REST API Specification [39].
3. Ogni risorsa può avere differenti rappresentazioni a seconda delle preferenze dell'utente. Sono supportati tutti i linguaggi di serializzazione RDF esistenti. L'unico formato non ancora supportato è l'html.
4. Non è stato ancora implementato il referenziamento indiretto per la risorsa non informativa sviluppatore, necessario in quanto, secondo le linee guida, le risorse non informative non possono essere accedute direttamente bensì solo tramite 303 redirect oppure Hash URI.
5. Content negotiation implementata grazie alla quale LiSeD fornisce una risorsa nel formato richiesto dal client all'interno della richiesta http inoltrata, come specificato nel campo "Accept" (pre esempio "Accept: application/rdf+xml).
6. URI aliases. Gli alias vengono ricercati dinamicamente e specificati tramite RDF link di tipo owl:sameAs.
7. Associated descriptions. Tutte le risorse, informative e non, sono associate a descrizioni di se stesse (metadata) in RDF.
8. Non viene mai utilizzata la reificazione di triple in RDF
9. Il container RDF "bag" viene utilizzato unicamente nelle richieste inoltrate per ottenere elenchi di URI di risorse. Di conseguenza non influenzano negativamente le ricerche Sparql in quanto le ricerche sono effettuate sulle singole risorse escludendo qualunque elencazione.
10. Per quanto possibile i vocabolari pre-esistenti sono stati riutilizzati.

11. Le risorse esterne dinamicamente collegate ai Linked Sensor data sono anch'esse tutte in formato Linked Data e, quindi, anch'esse collegate ad altre risorse e associate ad URI dereferenziabili.
12. Il vocabolario Hackstat
 - a. non è stato creato da zero, bensì è complementare ad altri vocabolari pre-esistenti.
 - b. è stato definito utilizzando l'RDF Schema.
 - c. I termini sono tutti identificati da URI http dereferenziabili. L'URI che identifica il vocabolario è esso stesso dereferenziabile
 - d. Essendo complementare ad altri vocabolari, sono stati integrati al suo interno termini e classi definiti da altri in altri vocabolari
 - e. I vincoli sono stati ridotti al minimo per fornire flessibilità ad eventuali espansioni future

5.3 ESPERIMENTI

Nel mostrare le modalità di interazione di LiSeD all'interno dei precedenti paragrafi, sono state effettuate e riportate le prove di esecuzione durante l'utilizzo di RESTful API, GUI, Java Client e Sparql. Inoltre sono stati effettuati JUnit test sul funzionamento del servizio di creazione Linked Data riguardanti sviluppatori, progetti ed Issue, tutti terminati con successo.

Sono stati creati degli ANT task che richiamano l'esecuzione dei tool Checkstyle, coverage, findbugs, etc. sull'implementazione di LiSeD realizzata.

Infine per non limitarsi all'appuramento del corretto funzionamento sintattico nella creazione dei Linked Data, ma verificando l'efficacia semantica in particolare dei collegamenti dinamicamente creati, sono stati effettuati degli esperimenti ottenendo i seguenti risultati.

Un progetto cui sono stati assegnati i Tag spam, e-mail, mailing_list, filter; è stato dinamicamente collegato tramite RDF link di tipo seeAlso con i seguenti Linked Data esterni:

- **GNU Mailman** - <http://rdfohloh.wikier.org/project/49> - "GNU Mailman is software to help manage email discussion lists and e-newsletters"
- **DSPAM** - <http://rdfohloh.wikier.org/project/4274> - "DSPAM is a scalable and open-source content-based spam filter designed for multi-user enterprise systems."
- **Thunderbird** - <http://rdfohloh.wikier.org/project/36> - "Thunderbird delivers. Enjoy safe, fast, and easy email, with intelligent spam filters, quick message search, and customizable views."

- **Greylist** - <http://rdfohloh.wikier.org/project/7222> - “Greylisting implementation for Postfix. Greylisting is a new method of blocking significant amounts of spam at the mailserver level.”
- **Claws Mail** - <http://rdfohloh.wikier.org/project/304> - “Claws Mail is a GTK+ based, lightweight, and fast email client.”

Un utente/sviluppatore associato all’indirizzo e-mail myriam.leggieri_at_gmail.com è stato dinamicamente collegato tramite RDF link di tipo sameAs con i seguenti Linked Data esterni:

- Profilo FOAF - <http://myr.netsons.org/foaf.rdf#iammyr>
- Profilo su RDFSOhloh - <http://rdfohloh.wikier.org/user/52334>
- Profilo su SiocLog - <http://irc.sioc-project.org/users/iammyr#user>

Un issue associato ai tag: browser, mysql, gui, avente ID 112 è stato dinamicamente collegato tramite RDF link di tipo sameAs al Linked Data <http://localhost:9875/linkedservicedata/issues/101> che, se visitato tramite browser, presenta la seguente descrizione RDF:

```
<http://localhost:9875/linkedservicedata/issues/101>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://xmlns.com/foaf/0.1/Defect> ;
  <http://commontag.org/ns#tagged>
    "mysql" , "browser" , "gui" ;
  <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/closedTime>
    "2009-06-20T00:00:00.000+02:00" ;
  <http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/modifiedTime>
    "2009-05-07T00:00:00.000+02:00" ;
  <http://dublincore.org/documents/dcmi-terms/#terms-isPartOf>
    "http://dasha.ics.hawaii.edu:9875/linkedservicedata/vocab/void" ;
  <http://www.w3.org/2002/07/owl#sameAs>
    "http://localhost:9875/linkedservicedata/issues/101";
  <http://xmlns.com/foaf/0.1/assigned_to>
    "http://localhost:9875/linkedservicedata/users/holatest@hackystat.org" ;
  <http://xmlns.com/foaf/0.1/created>
    "2006-10-12T00:00:00.000+02:00" ;
  <http://xmlns.com/foaf/0.1/duplicate>
    "http://localhost:9875/linkedservicedata/issues/112" ;
  <http://xmlns.com/foaf/0.1/name>
    "101" ;
  <http://xmlns.com/foaf/0.1/priority>
```



```
"high" ;  
<http://xmlns.com/foaf/0.1/status>  
"checked" ;  
<http://xmlns.com/foaf/0.1/target_milestone>  
"milestone1" .
```

Quindi sono stati collegati due issue caratterizzati dagli stessi tag.

In conclusione i collegamenti creati sono tutti pertinenti e, di conseguenza, il risultato è soddisfacente. Tuttavia la quantità di prove effettuate è ridotta e la sperimentazione verrà approfondita in futuro.

CAPITOLO VI

CONCLUSIONI E SVILUPPI FUTURI

Il servizio sviluppato si è rivelato in grado di fornire utili informazioni agli sviluppatori durante la loro attività di lavoro su progetti software. Attualmente però i vincoli imposti da Hackstat sulla privacy sono talmente rigidi da limitare eccessivamente l'accesso al dataset Hackstat pubblicato tramite LiSeD. I livelli di privacy costituiscono l'unico ostacolo ad un semplice accesso ai Linked Sensor Data di Hackstat, che pure è completamente supportato dalle caratteristiche di LiSeD quali Sparql endpoint e REST API. In tal modo ci si limita a sfruttare i dati altrui condivisi tramite Linked Data senza ricambiare con lo stesso spirito di condivisione. Lo sviluppo futuro più urgente quindi, anche in un'ottica di pieno ingresso e supporto al Web of Data, consiste nel rendere personalizzabili i livelli di privacy, sfruttando i meccanismi non invasivi suggeriti da LiSeD quali l'utilizzo di flag che indichino il livello preferito dall'utente a scelta tra: accesso consentito a chiunque; accesso consentito solo a se stessi e all'amministratore; accesso consentito solo ad utenti Hackstat.

Un'utile sviluppo futuro consiste anche nell'integrazione in Hackstat di un sistema che consenta di inviare messaggi istantanei o meno, agli sviluppatori che si necessita contattare, magari precedentemente filtrati in base alle competenze grazie alle funzionalità di LiSeD. In tal modo si evita di impiegare tempo nell'aprire un servizio di gestione posta o un social network o un sistema di messaggistica istantanea, contribuendo a creare uno spirito comunitario tra gli utenti Hackstat.

I Linked Sensor Data creati potrebbero essere utilizzati in futuro anche per altri scopi, quali per esempio la ricerca di persone da assumere per un impiego da parte di job recruiters; oppure la ricerca di componenti da integrare nel proprio sistema da parte di altri sviluppatori; casi d'uso originariamente proposti nel Google Summer of Code application form. Può essere utile anche l'aggiunta in LiSeD di una funzionalità grafica che permetta l'aggregazione di Linked Data provenienti da diverse sorgenti, per permettere agli sviluppatori una facile creazione di un nuovo tipo di Curriculum Vitae, contenente informazioni di garantita veridicità in quanto collezionati direttamente da sensori.

Infine per snellire l'applicazione e uniformarla al servizio ProjectBrowser è auspicabile un'interfaccia utente web-oriented piuttosto che l'attuale standalone, oppure è altrettanto auspicabile l'utilizzo di interfacce più immediate e innovative quali gli ambient device (Orb, Nabaztag, etc.).

BIBLIOGRAFIA

- [1] Jim McCarthy (1995). *Dynamics of Software Development*.
- [2] Dan Conde (2002). *Software Product Management: Managing Software Development from Idea to Product to Marketing to Sales*.
- [3] Robert K. Wysocki (2006). *Effective Software Project Management*.
- [4] World Wide Web Consortium. <http://www.w3.org/>
- [5] La bolla speculativa dot-COM. <http://www.einvestimenti.it/2008/11/la-bolla-speculativa-dot-com/>
- [6] Ruthfield, Scott (1995). *The Internet's History and Development From Wartime Tool to the Fish-Cam*.
- [7] Jolie O'Dell (2009). User Data Easier Than Ever to Phish on Facebook, New Study Shows. http://www.readwriteweb.com/archives/facebook-phishing-personal-data-privacy.php?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+readwriteweb+%28ReadWriteWeb%29&utm_content=Google+International
- [8] Tim O'Reilly (2005). What is Web 2.0. <http://oreilly.com/pub/a/web2/archive/what-is-web-20.html?page=1#mememap>
- [9] Sturla Bakke (2009). *Web 2.0 vs. Semantic Web: a socio-technical view based on information infrastructures research*.
- [10] Architettura web semantico (2005). http://it.wikipedia.org/wiki/Immagine:Architettura_web_semantico.png.
- [11] Herman, Ivan. Semantic Web Activity Statement (2008). <http://www.w3.org/2001/sw/Activity.html>.
- [12] Tom Gruber. What is an Ontology (1992). <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html>.
- [13] La rappresentazione della conoscenza nel Web Semantico: Architettura del web semantico. http://www.elearninglab.eu/studying/sw/sw_semantic_web.html.
- [14] La rappresentazione della conoscenza nel Web Semantico: Il web attuale e i motori di ricerca. http://www.elearninglab.eu/studying/sw/sw_web_today.html.
- [15] L. C. Fernández, R. Martínez-Tomás. *The Problem of Constructing General-Purpose Semantic Search Engines*.
- [16] Vocabulary of Interlinked Datasets: <http://rdfs.org/ns/void-guide>.
- [17] Michael Hausenblas. Nodalities Magazine (Sett./Ott. 2008). *Discovery and Usage of Linked Datasets on the Web of Data*.

- [18] R. Cyganiak, R. Delbru, G. Tummarello. Semantic Web Crawling: A Sitemap Extension (2007). <http://sw.deri.org/2007/07/sitemapextension/>.
- [19] Introduction to Umbel (2009). <http://umbel.org/intro.html>.
- [20] J. Abbate (1999). *Inventing the Internet*.
- [21] O. Hanseth, M. Aanestad (2002). *Bootstrapping networks, infrastructures and communities*.
- [22] Tim Berners Lee. Linked Data (2006). <http://www.w3.org/DesignIssues/LinkedData.html>.
- [23] O. Signore. RDF per la rappresentazione della conoscenza. <http://www.w3c.it/papers/RDF.pdf>.
- [24] AA. VV. Uniform Resource Identifiers (URI): Generic Syntax. <http://www.ietf.org/rfc/rfc2396.txt>.
- [25] D. Connolly, Addressing Schemes. <http://www.w3.org/Addressing/schemes>.
- [26] D. Bianchi. Introduzione a RDF (Resource Description Framework). http://www.ce.unipr.it/people/bianchi/Teaching/IntelligenzaArtificiale/WebSemantico_Ontologie/ResourceDescriptionFramework.pdf.
- [27] M. Picarelli. Interrogare l’RDF con SPARQL (2006). http://lau.csi.it/realizzare/accessibilita/linguaggi_programmazione/SPARQL/rdf.shtml.
- [28] J. Melton. SQL, XQuery, and SPARQL: What's Wrong With This Picture? (2006). <http://xtech06.usefulinc.com/schedule/detail/119>.
- [29] D. Beckett, J. Broekstra. SPARQL Query Results XML Format (2008). <http://www.w3.org/TR/rdf-sparql-XMLres/>.
- [30] E. Prud'hommeaux, A. Seaborne. SPARQL Query Language for RDF (2008). <http://www.w3.org/TR/rdf-sparql-query/>.
- [31] Tim Berners Lee. Linked Data. TED 2009 Conference (2009). <http://www.w3.org/2009/Talks/0204-ted-tbl/#%2822%29>.
- [32] T. Heath. Where is the business value in Linked Data (2008). <http://tomheath.com/blog/2008/09/where-is-the-business-value-in-linked-data/>.
- [33] Hackystat, Tutorial Guided Tour (2008). http://code.google.com/p/hackystat/wiki/Tutorial_GuidedTour.
- [34] H. Hamza. *A foundation for building stable analysis patterns* (2002).
- [35] T. Hastrup, U. Bojars, J. G. Breslin. SicoLog: providing IRC discussion logs as Linked data (2009). <http://tuukka.iki.fi/2009/sdow/sioclog-paper.pdf>.
- [36] FOAF-QDOS Beta – Reverse search. <http://foaf.qdos.com/reverse>.
- [37] S. Fernández. RDFOhloh (2008). <http://www.wikier.org/blog/rdfohloh>.
- [38] C. Bizer, R. Cyganiak e T. Heath. How to publish Linked Data on the Web (2007). <http://www4.wiwiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/>.

[39] M. Leggieri. REST API Specification (2009). *http://code.google.com/p/hackystat-linked-sensor-data/wiki/RestApiSpecification_NEW*.