Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Corso di Laurea Magistrale in Informatica

Uno strumento di supporto all'analisi e visualizzazione di dati strutturati nel contesto giuridico

Relatore:
Chiar.mo Prof. Fabio Vitali

Presentata da:
Federico Giubaldo

Correlatori:
Chiar.ma Prof.ssa Monica Palmirani
Dott. Luca Cervone

Anno Accademico 2017-18
Sessione II

"Lottate, ragionate col vostro cervello, ricordate che ciascuno è qualcuno, un individuo prezioso, responsabile, artefice di se stesso, difendetelo il vostro io, nocciolo di ogni libertà, la libertà è un dovere, prima che un diritto è un dovere."

Oriana Fallaci

Contenuti

Elenco delle figure

Capitolo 1

Introduzione

Il mio interesse e l'obiettivo di questa dissertazione è di mettere insieme la Data Analysis e l'Informatica Giuridica analizzando i risultati già ottenuti e le potenzialità della loro interazione. A tal fine, la dissertazione mira a dimostrare che *la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.*

Oggigiorno viene prodotta una quantità ingente di dati digitali provenienti da una moltitudine di sorgenti; tale situazione ha dato una nuova e importante spinta agli ambiti relativi all'analisi dei dati che mirano al miglioramento e allo sviluppo di servizi, all'identificazione di nuovi trend, alla comprensione degli andamenti di un certo contesto, al supporto delle decisioni, etc. Negli ultimi decenni è emerso anche un notevole interesse verso l'informatizzazione degli enti istituzionali, sia nell'automazione delle attività che nella digitalizzazione dei documenti prodotti. Quest'ultimo aspetto ha reso accessibili e fruibili i documenti giuridici attraverso gli elaboratori aprendo le porte a una moltitudine di opportunità prima inimmaginabili. Infatti tale situazione ha permesso lo sviluppo di strumenti e meccanismi per la creazione di nuove conoscenze e di nuovi servizi che mirano a sfruttare il grande valore sociale ed economico che si nasconde all'interno delle grandi quantità di dati.

Sulla base di tale contesto culturale e tecnologico, il lavoro di tesi che ho sviluppato si focalizza su due discipline ben diverse, entrambe rilevanti sia da

un punto di vista accademico che da un punto di vista sociale ed economico, e sulla loro interazione: Data Analysis e Informatica Giuridica.

La Data Analysis è la disciplina che permette di raccogliere, utilizzare e analizzare i dati in modo da evidenziare specifici fenomeni. Spesso questi ultimi sono successivamente resi accessibili e comprensibili agli utenti finali tramite degli strumenti di Data Visualization. L'avanzare dello sviluppo tecnologico, il numero crescente di dispositivi sempre più potenti a prezzi sempre più accessibili, e l'espansione degli applicativi e dei servizi orientati all'utente hanno fatto registrare un incremento significativo dei dati digitali prodotti quotidianamente. Tale situazione ha portato notevole interesse verso la Data Analysis al fine di beneficiare di queste ingenti quantità di dati.

L'Informatica Giuridica si occupa di svariati aspetti connessi all'interazione tra informatica e diritto, infatti, essa individua due campi di applicazione: informatica del diritto e diritto dell'informatica. Questa dissertazione si sofferma sull'informatica del diritto e, in particolare, sulla standardizzazione dei documenti giuridici; tale aspetto si occupa della definizione di standard per la rappresentazione digitale dei documenti giuridici che garantiscono un insieme di proprietà fondamentali. Gli standard rendono possibile la navigazione, l'analisi e l'estrazione di informazioni dai documenti giuridici attraverso un elaboratore. Un aspetto importante in questo ambito è la possibilità di aggiungere, all'interno dei documenti, informazioni semantiche che non alterano il valore legale del documento ma aprono le porte ad analisi sempre più complesse, sofisticate e precise.

Una parte del lavoro che ha permesso la stesura di questa dissertazione è stata realizzata all'interno del CIRSFID[1] (Centro Interdipartimentale di Ricerca in Storia del Diritto, Filosofia e Sociologia del Diritto e Informatica Giuridica), centro interdisciplinare dell'Università di Bologna. Un'area rilevante all'interno del CIRSFID è quella dell'Informatica Giuridica, la quale vanta numerosi progetti di ricerca con diversi enti, sia nazionali che mondiali.

Allo scopo di dimostrare la tesi della dissertazione si è sviluppato dapprima un attento studio dello stato dell'arte presentando il processo della Data Analysis, inclusa la Data Visualization, e l'importanza che riveste nel mondo di oggi. In seguito si è approfondito il concetto di Big Data e le necessità

[1] `http://http://www.cirsfid.unibo.it/`

che hanno portato alla sua nascita, come le grandi quantità di dati prodotte quotidianamente. Questo percorso conduce alla trattazione dei dati pubblici, discutendo del valore sociale ed economico che può emergere dalla loro analisi. In tale contesto viene discusso il concetto dei dati aperti e il modello Open Government Data come fattore che ha influenzato l'incremento dei dati pubblici aperti e accessibili. La grande quantità di dati disponibili in rete ha incentivato lo sviluppo e la nascita di nuovi strumenti e ambiti lavorativi come il Data Journalism e il DAF[2]. Il Data Journalism è l'ambito in cui si applicano i metodi scientifici alle pratiche del giornalismo in modo da dare a quest'ultimo una valenza e una robustezza scientifica. Il DAF è invece uno strumento italiano ancora in fase di sviluppo che mira a supportare le attività della pubblica amministrazione e a migliorarne i servizi offerti.

Successivamente viene introdotta l'Informatica Giuridica per arrivare a discutere dell'ambito relativo alla standardizzazione dei documenti giuridici. L'obiettivo principale di tale ambito è la definizione di standard, i quali consentono l'arricchimento dei documenti giuridici tramite elementi semantici; tali standard permettono di processare i documenti tramite un elaboratore e inoltre garantiscono certe proprietà fondamentali nel mondo giuridico come ad esempio la validità nel tempo. Per perseguire tale obiettivo è necessario introdurre i documenti strutturati, ovvero documenti in cui il testo può essere arricchito con elementi che specificano certi significati; i documenti strutturati possono essere realizzati tramite i linguaggi di markup, come ad esempio XML, che permettono la definizione di elementi nel testo.

A questo punto sono presentati alcuni esempi che mettono insieme l'ambito della Data Analysis con l'ambito dell'Informatica Giuridica al fine di mostrare alcuni risultati raggiunti e le potenzialità di tale cooperazione. Gli esempi presentati sono: Code4Italy, Socievole e il progetto "Network of French legal codes". Code4Italy è incentrato sull'analisi e sulla creazione di rappresentazioni semplici e intuitive di alcuni aspetti significativi del procedimento legislativo del Parlamento italiano. Socievole (SOCIal Evolution and Visualization Of Legal acts) utilizza i documenti della legislazione inglese al fine di creare una timeline che rappresenta l'evoluzione dei temi sociali attraverso la legislazione. Nel progetto "Network of French legal codes" vie-

[2] *Data & Analytics Framework Italia*, `https://dataportal.daf.teamdigitale.it/`

ne proposta un'analisi delle leggi francesi selezionando 52 codici legali che hanno permesso lo sviluppo di una rete in cui i codici rappresentano i vertici, mentre le citazioni tra i codici rappresentano gli archi. Questi esempi mostrano l'utilità di avere documenti giuridici in un certo standard in modo tale da permettere analisi complesse e specifiche tramite un elaboratore. Uno standard XML appropriato per tale contesto è *Akoma Ntoso*, il quale si discosta da altri standard per il suo carattere universale e la sua ricchezza semantica. *Akoma Ntoso* ha avuto un notevole successo diventando, oggi, uno standard OASIS sotto il nome di *LegalDocML*.

Sulla base di documenti *Akoma Ntoso* sono state definite alcune analisi e contestualmente sono stati sviluppati degli strumenti per la loro esecuzione e la rappresentazione grafica dei risultati ottenuti; gli strumenti realizzati sono: la dashboard *SOFIA* (Structured Open government data For Interactive Analysis) e la libreria *akomando-db*. *SOFIA* è uno strumento per l'esecuzione di analisi su collezioni di documenti e per la presentazione delle relative rappresentazioni grafiche. *Akomando-db* è invece una libreria JavaScript che offre un insieme di funzionalità per gestire le fasi di archiviazione e di recupero dei documenti *Akoma Ntoso*; tale libreria è inoltre una componente importante di *SOFIA*.

Questi due strumenti sono fondamentali nella costruzione di un percorso che ha l'obiettivo di confermare o confutare la tesi iniziale secondo cui la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization. Per tale verifica viene presa in considerazione la figura del data journalist al fine di comprendere e valutare il lavoro e lo sforzo necessario per l'analisi dei dati e la creazione di strumenti che permettono di comunicare i risultati ottenuti. Il Data Journalism è infatti un ambito che pone le sue fondamenta in diverse discipline, principalmente nel giornalismo ma anche nell'informatica e nella Data Analysis. Queste caratteristiche fanno del data journalist il punto di incontro ideale tra la Data Analysis e l'Informatica Giuridica e quindi l'utente di riferimento per la valutazione finale. Quest'ultima si basa sull'analisi di due differenti scenari relativi allo sviluppo di *SOFIA*: uno in cui il data journalist utilizza *akomando-db*; l'altro in cui viene discusso lo sviluppo di

SOFIA senza l'ausilio di *akomando-db*. Questo secondo caso mette in luce la complessità e il tempo necessario alla comprensione e all'utilizzo dei vari strumenti che hanno portato alla conferma della tesi iniziale secondo cui la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

La dissertazione si sviluppa come segue. Nel **capitolo 2** sarà trattata la Data Analysis e le fasi che la compongono (sezioni 2.1 e 2.2) per poi introdurre i Big Data (sezione 2.3). In seguito sarà analizzato il contesto degli enti istituzionali e come questo si stia muovendo in direzione degli Open Data, più precisamente degli Open Government Data (sezione 2.4.1). Infine si parlerà dei contesti applicativi della Data Analysis, come l'ambito del Data Journalism (sezione 2.4.2) e il Data & Analytics Framework (sezione 2.4.3).

Nel **capitolo 3** si introdurrà il lettore all'Informatica Giuridica (sezione 3.1) e sarà discusso l'interesse di tale disciplina verso la standardizzazione dei documenti legali (sezione 3.1.1). In seguito saranno descritti gli strumenti per arrivare a tale obiettivo, ovvero i linguaggi di markup e il linguaggio XML (sezioni 3.2.1 e 3.2.2).

Nel **capitolo 4** saranno presentati alcuni esempi di Data Analysis applicata all'Informatica Giuridica (sezione 4.1) e contestualmente si descriveranno gli standard XML che rendono possibili tali analisi sui documenti giuridici (sezione 4.2); per completare la trattazione degli standard XML saranno descritti gli strumenti basilari che favoriscono il loro utilizzo e la loro diffusione (sezione 4.2.1) oltre alla loro classificazione in diverse categorie (sezione 4.2.2). In seguito sarà introdotto lo standard *Akoma Ntoso* (sezione 4.3) e saranno approfonditi gli aspetti riguardanti la naming convention e il modello FRBR (sezione 4.3.3).

Con il **capitolo 5** inizierà la descrizione della parte relativa al progetto di tesi. In questo capitolo verranno definite alcune analisi sui documenti *Akoma Ntoso* e le motivazioni alla base della scelta di tale standard (sezione 5.1); in seguito verranno descritte le funzionalità della dashboard che ho sviluppato (sezione 5.2.1) e le componenti principali per il suo sviluppo (sezione 5.2.2). Infine verrà introdotta la libreria *akomando* (sezione 5.3) e si discuterà della mancanza di uno strumento per la gestione delle fasi di archiviazione e recu-

pero dei documenti, necessario per lo sviluppo di *SOFIA* e per l'utilizzo di *akomando* ai fini delle analisi (sezione 5.4).

Nel **capitolo 6** sarà descritto *akomando-db*, componente principale del mio progetto di tesi. Saranno trattati gli obiettivi e i casi d'uso (sezioni 6.1 e 6.2), in seguito sarà presentata l'API di e le funzionalità accessibili a partire dalle funzioni dell'API (sezione 6.3); infine si discuterà dell'architettura e dell'implementazione di tale libreria (sezione 6.4).

Nel **capitolo 7** saranno mostrati i risultati ottenuti dalle analisi eseguite sui documenti *Akoma Ntoso* attraverso i grafici della dashboard (sezione 7.1). Successivamente si cercherà di dare una risposta alla tesi della dissertazione attraverso una valutazione qualitativa relativa alla costruzione di *SOFIA* (sezione 7.2).

La dissertazione si concluderà con la presentazione di un riassunto del lavoro fatto e degli argomenti trattati, seguito da potenziali nuovi sviluppi per chi auspicabilmente prenderà ed espanderà il mio progetto di tesi.

Capitolo 2

Dati digitali: analisi e utilità in ambito istituzionale

In questa sezione si discuterà il concetto di Data Analysis, i fattori che hanno influenzato tale ambito e l'importanza che riveste oggi, in particolare, nel mondo degli enti istituzionali.

2.1 Cos'è la Data Analysis

La Data Analysis, oggi, riveste un ruolo di primo piano per le organizzazione, per le aziende e anche per gli enti istituzionali. Tale rilevanza è diventata più evidente grazie a diversi fattori come l'utilizzo sempre più intenso del web, l'incremento di dispositivi connessi in rete e la moltitudine di servizi messi a disposizione degli utente. Tutti questi fattori hanno inciso nella crescita della quantità di dati prodotta giornalmente in rete. Tale situazione ha reso i dati digitali una fonte preziosa per l'esecuzione di analisi che mirano ad estrapolare informazioni e tendenze che possono essere utilizzate come strumenti di supporto per le decisioni interne ad un'azienda o per il miglioramento dei servizi offerti [Eco10].

John Wilder Tukey, matematico americano del '900, definì la Data Analysis come l'insieme delle procedure per l'analisi dei dati, le tecniche per l'interpretazione dei risultati prodotti da tali procedure, i modi per la piani-

ficazione della raccolta dati, i meccanismi e risultati statistici che si applicano all'analisi dei dati [Tuk62].

Il processo di Data Analysis, in accordo con O'Neil e Schutt [OS13], può essere visto come un insieme di fasi così riassunte:

- **Data Collection**: fase riguardante la raccolta di dati che può avvenire da differenti sorgenti;
- **Data Processing**: molto spesso i dati collezionati non sono pronti per essere analizzati, quindi risulta inevitabile una fase di elaborazione che abbia l'obiettivo di organizzarli in una struttura tale da poter essere utilizzata per le fasi di analisi;
- **Data Cleaning**: fase in cui si cerca di armonizzare i dati raccolti in quanto essi possono essere incompleti o presentare duplicati;
- **Exploratory Data Analysis**: fase che aiuta nella formulazione di ipotesi riguardanti il fenomeno che si vuole analizzare e nella selezione degli strumenti e delle tecniche da utilizzare. Inoltre, tale fase può suggerire ulteriori step di armonizzazione dei dati o anche di raccolta dati;
- **Confirmatory Data Analysis**: applicazione di determinate tecniche al fine di trasformare i dati analizzati in informazioni che permettano di verificare o confutare le assunzioni fatte nella fase precedente;
- **Data Visualization**: fase che ha come obiettivo la visualizzazione dei risultati ottenuti, al fine di renderli accessibili e facilmente fruibili a tutti.

La figura 2.1 mostra i passi appena discussi tramite una rappresentazione grafica, partendo dal mondo reale dove le persone sono la sorgente primaria di dati; quest'ultimi vengono prodotti direttamente o indirettamente durante le attività svolte dagli individui. A questo punto vi sono i dati grezzi che possono provenire da una moltitudine di sorgenti e che, di per sé, non danno nessuno informazione a chi li osserva. La prima fase, una volta collezionati i dati, è il processamento di quest'ultimi al fine di organizzarli in un certo formato che sarà utilizzato per le fasi di analisi. Dopo la fase di Data Processing

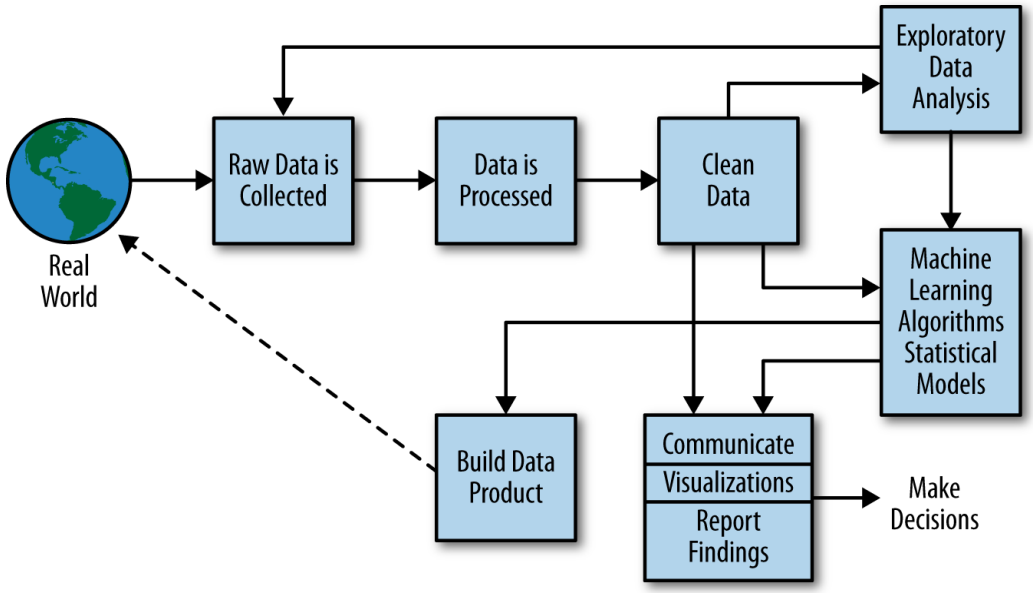

Figura 2.1: Fasi della Data Analysis

vi è la fase di pulizia dei dati ovvero dove questi vengono armonizzati rimuovendo i duplicati e correggendo le informazioni mancanti. Segue una fase di exploratory Data Analysis (EDA) che può risultare in ulteriori passaggi delle fasi già eseguite. Una volta che i dati sono organizzati seconda in una certa struttura e sono armonizzati, si può procedere con l'applicazione degli algoritmi; la scelta di quest'ultimi dipende dal tipo di problema trattato e dalle ipotesi che si vogliono verificare. I risultati ottenuti possono essere riportati attraverso vari tipi di strumento come, ad esempio, le visualizzazioni attraverso i grafici.

In letteratura è possibile trovare diverse concettualizzazioni del processo di Data Analysis. Ad esempio, si può fare una distinzione tra l'approccio lineare e l'approccio ciclico: quello lineare è caratterizzato dalla rigidità della struttura e dell'organizzazione delle fasi; l'approccio ciclico, discusso sopra, è un approccio caratterizzato dalla flessibilità infatti permette di tornare ad una fase precedente per eseguirla nuovamente al fine di trarne benefici [Ric06].

2.2 Fasi della Data Analysis

Si è brevemente accennato alle fasi che compongono il processo di Data Analysis; tali fasi verranno approfondite di seguito.

2.2.1 Data Collection, Processing e Cleaning

Le fasi di data collection, data processing e data cleaning si occupano della raccolta, del filtraggio, della formattazione e della pulizia dei dati, prima che questi vengano passati alle fasi di analisi.

I dati vengono collezionati a partire, potenzialmente, da una moltitudine di sorgenti quali i social networks, i sensori connessi in rete(telecamere, satelliti, microfoni), applicativi per l'acquisto e la vendita di oggetti, applicazioni per i telefonini, servizi offerti dalle pubbliche amministrazioni, e da numerose altre sorgenti che fanno dei dati il loro bene primario.

L'organizzazione dei dati è affidata alla fase di data processing che ha il compito di strutturarli in un certo formato, facendo attenzione alla perdità di dati rilevanti durante l'elaborazione, al fine di utilizzare determinate tecnologie che possono avere restrizioni riguardanti il formato dei dati in ingresso [OS13].

I risultati della Data Analysis possono essere profondamente influenzati dalla presenza di dati inconsistenti o scorretti, questo è uno scenario che si vuole assolutamente evitare poiché renderebbe inutilizzabili i risultati ottenuti. La Data Cleaning, detta anche Data Cleansing, si pone l'obiettivo di mettere a disposizione dell'utente meccanismi e procedure per l'individuazione automatica o semiautomatica degli errori presenti nei dataset, ed eventualmente di strumenti per la loro correzione [RD00].

2.2.2 Exploratory e Confirmatory Data Analysis

John Wilder Tukey affermava che ne l'exploratory Data Analysis(EDA) ne la confirmatory Data Analysis(CDA) sono sufficienti se usate singolarmente; queste due fasi possono e devono essere utilizzate insieme, prima la EDA e poi la CDA [Tuk80].

IN accordo con lo studio di Behrens [Beh97] l'obiettivo della exploratory Data Analysis è quello di formulare ipotesi e cercare dei pattern sui dati al fine di delineare dei modelli plausibili. Invece il ruolo della confirmatory Data Analysis è quello di testare tali ipotesi e tali modelli, attraverso approcci probabilistici, al fine di verificarne la correttezza. Queste due fasi sono cicliche tra loro in quanto vi è un continuo tra la formulazione di nuove ipotesi e la loro verifica.

Un'analogia interessante è quello di rappresentare l'exploratory Data Analysis come un interrogatorio dove vengono presentate storie corrette e storie corrotte, mentre la confirmatory Data Analysis come un insieme di prove per tali storie nate a partire da procedure standard e consolidate.

2.2.3 Data Visualization

La data visualization è la fase del processo di Data Analysis che si occupa della presentazione, tramite visualizzazioni grafiche, dei risultati ottenuti e della comunicazione di quest'ultimi. L'obiettivo principale è quello di far emergere il valore dei dati analizzati e permettere una facile comprensione e interpretazione dei risultati ottenuti.

In accordo con il lavoro di Khan [KK11], la data visualization può essere descritta come un processo che comprende sei fasi e che ha lo scopo di creare visualizzazione con un design preciso e con un numero di errori minimo. Tali fasi possono essere riassunte nel modo seguente:

- **Mapping**: il focus di questa fase è quello di decidere come visualizzare le informazioni, ovvero come codificarle dentro una rappresentazione grafica;
- **Selection**: si occupa della selezione dei dati da visualizzare; tale scelta è strettamente dipendente dall'obiettivo che si vuole raggiungere attraverso le visualizzazioni grafiche;
- **Presentation**: gestione della struttura e dell'organizzazione relativa alle informazioni da presentare nello spazio di lavoro dell'utente finale;
- **Interactivity**: gestione delle funzionalità messi a disposizione dell'utente al fine di organizzare, esplorare e riorganizzare le visualizzazioni;

- **Usability**: fase concernente il campo dell'interazione uomo-macchina. Tale fase ha l'obiettivo di migliorare l'usabilità delle visualizzazioni e intervenire al fine di aumentare la soddisfazione complessiva dell'utente finale;

- **Evaluation**: quest'ultima fase ha come scopo quello di capire se le tecniche di visualizzazione utilizzate sono efficaci o meno, verificare l'effettivo raggiungimento dell'obiettivo posto e, in generale, valutare complessivamente le rappresentazioni grafiche sviluppate.

Con il passare degli anni sono state sviluppate un numero sempre più elevato di strumenti per la Data Visualization che vanno in direzione dei principi emersi dalle fasi trattate precedentemente: usabilità, interattività, funzionalità offerte, etc. La varietà di strumenti disponibili permette agli utilizzatori di poter scegliere quelle più appropriate all'obiettivo che si intende raggiungere. Tra gli strumenti più comunemente utilizzati troviamo: le tabelle, i grafici a torta, i grafici a linee, i grafici a barre, gli istogrammi, i grafici a dispersione, etc.

2.3 Big Data

Oggi, nel mondo, vi è un'enorme quantità di dati digitali che sta diventando sempre più vasta, sempre più velocemente. Questo ha dato vita ad una moltitudine di opportunità che prima erano impensabili come ad esempio la prevenzione di malattie[1] e la lotta al crimine[2]. Analizzando le motivazione che stanno alla base di tale incremento dimensionale non si può che parlare dello sviluppo tecnologico. Tale sviluppo ha portato sul mercato dispositivi digitali sempre più potenti e con prezzi sempre più accessibili, come i vari sensori che adesso ci circondano e che permettono di portare in digitale molte informazioni che prima non erano disponibili. [Eco10] Tutto questo ha fatto nascere l'esigenza di adeguare le tecnologie e i database tradizioni alle

[1] www.agendadigitale.eu/sanita/big-data-in-sanita-perche-sono-promessa-mancata-ecco-la-svolta-necessaria

[2] *Crimespotting*, `https://stamen.com/work/crimespotting/`

necessità di un mondo in cui si deve lavorare con grandi quantità di dati, collezionati in modo continuo e che possono essere eterogenei tra loro.

Queste esigenze sono alla base del concetto di "big data", concetto che è stato al centro di molti lavori e studi nell'ultimo decennio e che porta con sé varie sfumature per quanto riguarda la sua descrizione. Tali sfumature possono essere riassunte associando i big data con l'identificazione di enormi dataset che raccolgono dati proveniente da diverse sorgenti e sono analizzati tramite tecnologie e algoritmi specifici. Le peculiarità che caratterizzano i big data possono essere riassunte nei seguenti quattro punti:

- **Volume**: enormi quantità di dati da processare; solamente Facebook, nel 2015, gestiva 500 terabyte di nuovi dati al giorno;

- **Velocità**: i dati devono poter essere analizzati in tempo reale, o comunque in modo abbastanza veloce, così da gestire il flusso di dati in entrata senza far diventare il dato obsoleto; ad esempio, i continui flussi di dati proveniente dai social network oppure i sensori che producono incessantemente informazioni relative alle loro funzioni;

- **Varietà**: i dati possono provenire da sorgenti differenti ed essere caratterizzati da diversi formati e tipologie di dato; ad esempio possono essere strutturati e non strutturati: dati numerici, geospaziali, immagini, file video, testi non strutturati (es. file di log), etc.;

- **Veridicità**: caratteristica riguardante la qualità e l'accuratezza dei big data; descrive l'incompletezza dei dati che è caratterizzata dalla loro inconsistenza e dalla loro inaccuratezza.

Per quanto riguarda l'utilizzo e l'utilità dei big data, spesso non ci si rende conto di come, ormai, siano alla base di molti servizi e applicazioni. Se pensiamo, ad esempio, alle informazioni che vengono inviate giornalmente dagli smartphone, relativamente ai luoghi visitati e ai percorsi fatti, che ci permettono di ricevere informazioni sui posti di interesse limitrofi; oppure alle quantità di dati proveniente da sensori e telecamere che le nuove auto senza guidatore devono processare al fine di garantire certi standard di sicurezza; un ulteriore esempio, ormai entrato nella vita quotidiano, sono i suggerimenti

offerti da applicazioni di intrattenimento o smart Tv, che si basano sulle nostre preferenza, le quali vengono derivate dalle analisi su grandi quantità di dati che noi stessi forniamo in modo diretto o indiretto [DDK15] [LJ12].

2.4 Contesti applicativi della Data Analysis

Nelle sezioni precedenti si è discusso di come l'incremento della quantità di dati sia stato negli ultimi decenni incessante e, ancora oggi, in continua crescita. Parallelamente a questo trend, si è visto il grande interesse verso lo sviluppo di tecniche e strumenti che permettessero di archiviare, processare e analizzare queste ingenti moli di dati in modo efficace. Un altro aspetto interessante è l'incremento dell'interesse da parte di aziende, organizzazioni, enti instituzionali e ambiti professionali verso l'utilizzo e l'analisi dei dati al fine di ottenere risultati che possano essere una risorsa su cui organizzare strategie, prendere decisione o migliorare i servizi. Esempi di tale situazione sono rappresentati dal Data Journalism, trattato nella sezione 2.4.2 e dal DAF che sarà trattato nella sezione 2.4.3.

Esistono molti progetti interessanti che permettono di mostrare come l'analisi dei dati, insieme a precise tecniche di Data Visualization, possa essere alla base di servizi utili e innovativi. Una società che lavora in questa direzione è la Stamen Design LLC[3], fondata da Eric Rodenbeck a San Francisco nel 2001 che si occupa di Data Analysis, Data Visualization e strategie per la Data Communication.

Uno dei tanti progetti sviluppati dalla Stamen Design LLC è Crimespotting[4], avviato nelle città di San Francisco e Oakland. Tale progetto dà la possibilità di ricevere informazioni, tramite differenti canali comunicativi, sugli aggiornamenti dei crimini accaduti in certe zone della città. Uno di questi canali comunicativi è la mappa interattiva, come mostrato nella figura 2.2, che permette in modo dinamico di visualizzare i punti in cui vi sono stati dei crimini oltre al tipo di crimine commesso [Eco10].

In questa breve introduzione si è accennato al concetto di dati pubblici, argomento approfondito nella sezione 2.4.1, e di come questi possono esse-

[3] *Stamen Design LLC*, `https://stamen.com/`

[4] *Crimespotting*, `https://stamen.com/work/crimespotting/`

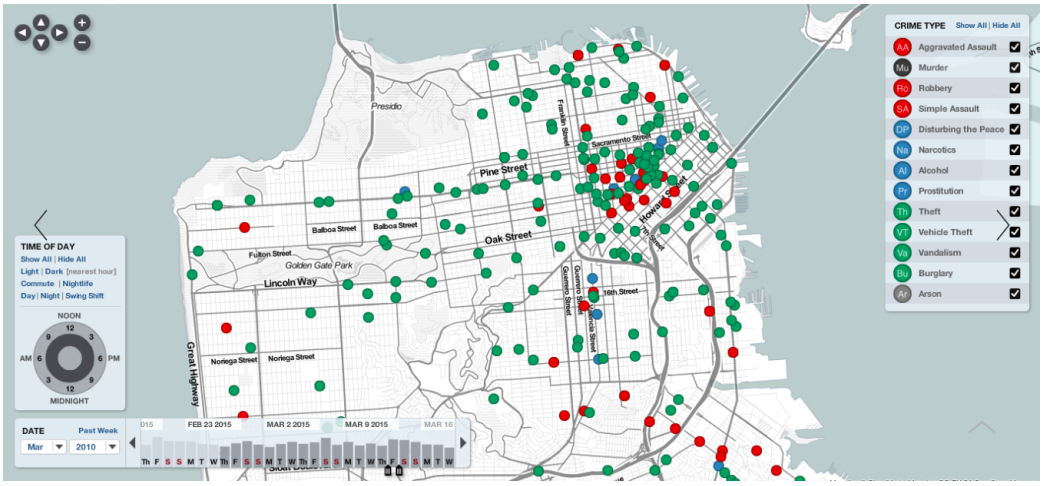

Figura 2.2: Mappa Interattiva Crimespotting

re utilizzati per eseguire analisi e creare visualizzazioni come ad esempio nell'ambito del Data Journalism (sezione 2.4.2). Uno strumento di grande rilevanza per l'esecuzioni di analisi, nato e tuttora in fase di sviluppo in Italia, è il DAF che sarà esposto nella sezione 2.4.3.

2.4.1 Verso il mondo dei dati pubblici aperti

Come si evince dal progetto "Crimespotting", i dati pubblici come quelli riguardanti la criminalità sono una risorsa importante se resi disponibili e quindi utilizzabili per la Data Analysis al fine di ottenere dei benefici che possono concretizzarsi in diversi contesti, come visto precedentemente.

L'idea di rendere aperti e accessibili i dati pubblici, oggi, è consolidata in molte nazioni. Tale processo di apertura e condivisione è stato supportato e incentivato anche dall'avvento dell'open government. Questo modello per gli enti istituzionali, infatti, mette in primo piano i cittadini, l'informazione e le conoscenze derivate a partire da quest'ultima. Il modello open government è caratterizzato dall'utilizzo delle tecnologie informatiche che permettono di migliorare l'informazione e i servizi offerti, incoraggiare la partecipazione dei cittadini e rendere i governi più responsabile, trasparente e efficace.

Un'interessante movimento, cresciuto sotto l'influenza dell'open government, è l'Open Government Data (OGD) che ha avuto inizio nel 2009 in

U.S e U.K. Tale movimento nasce per promuovere la partecipazione, la trasparenza e la cooperazione attraverso la pubblicazione e la divulgazione dei dati provenienti dalle pubbliche amministrazioni. Rendendo aperti e accessibili tali dati a tutti i cittadini e agli enti interessati si vuole fornire uno strumento di supporto per la creazione di business e di servizi innovativi che forniscano un valore sociale e commerciale al contesto in cui vengono applicati [PMG14]. E' importante sottolineare come l'Open Government Data, non inteso come movimento bensì come concetto concreto (dati aperti, creati da enti pubblici), rientra in un concetto più generale che è quello dell'open data, ovvero dati aperti che possono essere creato sia da enti privati che da enti pubblici [FP16].

Oggi si possono trovare moltissimi esempi che concretizzano l'idea dell'Open Government Data, alcuni di questi sono:

- il portale americano data.gov[5], nato nel 2009;
- il portale del Regno Unito, data.gov.uk[6], nato nel 2009;
- il portale italiano dati.gov.it[7], nato nel 2011.

La figura 2.3 mostra una mappa dove sono riportati i portali di Open Government Data presenti nel mondo[8].

Cosa si intende con "dati aperti". In precedenza si è parlato di dati aperti ma non si è discusso del significato di tale termine. Per poter parlare di dati aperti, in accordo con Ubaldi [Uba13], è necessario che i dati siano disponibili e accessibili da un lato, e riusabili e ridistribuibili dall'altro.

Per poter considerare i dati riusabili e ridistribuibili è necessario che:

- devono essere disponibili in un formato machine-readable ad esempio XML (eXtensible Markup Language), linguaggio trattato nella sezione 3.2.2;

[5] *Data.gov*, `https://www.data.gov/`
[6] *data.gov.uk*, `https://data.gov.uk/`
[7] *dati.gov.it*, `https://www.dati.gov.it/`
[8] Estratta dalla pagine `https://www.data.gov/open-gov/`

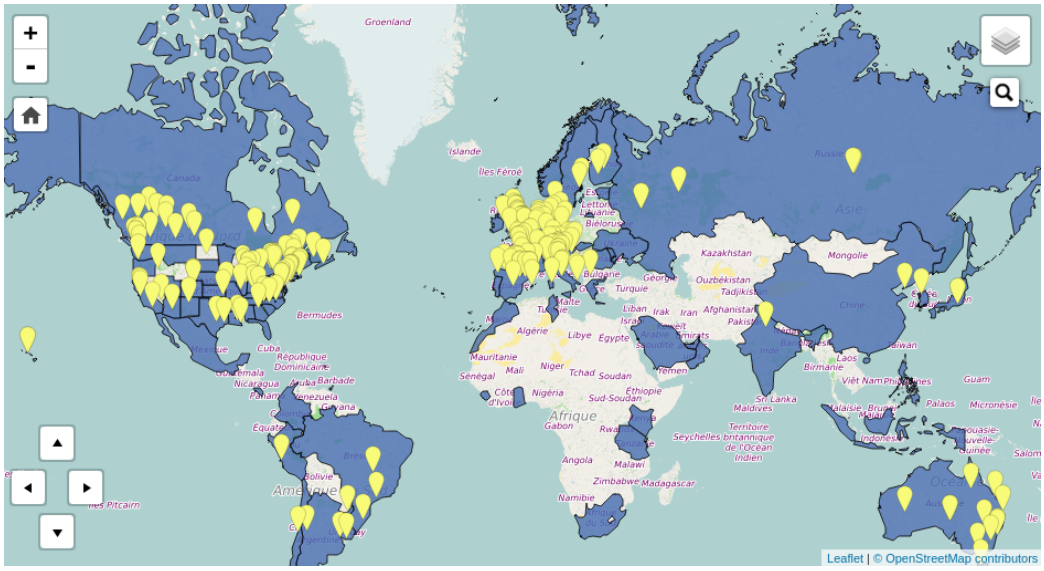

Figura 2.3: Mappa dei Portali di Open Government Data

- devono essere disponibili in un formato aperto, ovvero non limitato dall'utilizzo di specifici software;
- deve essere possibile scaricare interi dataset e non solamente alcune entry;
- devono essere rilasciati in modo tempestivo così da poter sfruttare il loro valore quando risulta ancora interessante e utile;
- devono poter essere riutilizzati senza restrizioni.

Invece, per poter considerare i dati disponibili e accessibili, essi devono essere offerti in modo gratuito oppure al costo della riproduzione e distribuzione, oltre che liberamente scaricabili attraverso la rete. Un ulteriore requisito per poter parlare di dati aperti è la necessità di prevedere e fornire degli strumenti che permettano in modo facile e veloce la loro individuazione e ricerca.

2.4.2 Il Data Journalism

Un settore che nell'ultimo decennio ha riscosso notevole interesse è quello del Data Journalism. Tale ambito, in accordo con Heravi [Her18], è caratteriz-

zato da una moltitudine di discipline che ne caratterizzano il background: giornalismo, scienze dell'informazione, scienze sociali, data scienze, informatica, Data Analysis e design dell'informazione. Il Data Journalism promuove un giornalismo che si basa su approcci scientifici, garantendo la trasparenza e la riproducibilià dei risultati ottenuti. I data journalist possono avere un background vario; dagli studi riportati nell'articolo di Heravi, le principali capacità sono nell'ambito del giornalismo, della Data Analysis, della programmazione software e della statistica. Inoltre Heravi mette in luce i due aspetti di maggior interesse per i data journalist che sono la Data Analysis e la programmazione software.

Esistono una moltitudine di strumenti a disposizione del Data Journalism[9] ma, come si evince dagli ambiti di interesse discussi precedentemente, uno strumento molto interessante e potente è dato dalla programmazione software e dal mondo delle API.

Un esempio di progetto nell'ambito del Data Journalism, vincitore del Data Journalism Award 2015[10], è "People's republic of Bolzano[11]". Tale progetto descrive, tramite una design accattivante e intuitivo, l'evoluzione della comunità cinese a Bolzano.

2.4.3 DAF: strumento per l'analisi dei dati pubblici in Italia

Il Data & Analytics Framework[12] (DAF) è uno strumento, attualmente in fase di sviluppo da parte del Team per la Trasformazione Digitale, che mira a valorizzare il patrimonio informativo pubblico italiano.

La trattazione di questa sezione segue il documento presentato dal Team Digitale[13] relativo al piano di sviluppo del DAF [AT18].

[9] `https://www.theguardian.com/news/datablog/2010/oct/01/data-journalism-how-to-guide`

[10] *Data Journalism Awards*,`https://www.datajournalismawards.org/`

[11] *People's republic of Bolzano*, `http://www.peoplesrepublicofbolzano.com/`

[12] *Data & Analytics Framework Italia*, `https://dataportal.daf.teamdigitale.it/`

[13] *Team per la Trasformazione Digitale*, `https://teamdigitale.governo.it/`

Obiettivi del DAF. Questo strumento si pone l'obiettivo di essere un punto di riferimento per la pubblica amministrazione (PA), offrendo un portale unico dove accedere ai dati prodotti dalle PA e da altre sorgenti connesse. Tali dati possono essere manipolati attraverso gli strumenti messi a disposizione nel portale del DAF.

Più in dettaglio l'idea è quella di aprire il mondo della pubblica amministrazione ai benefici offerti dalle moderne piattaforme per la gestione e l'analisi dei dati con l'obiettivo di:

- incrementare il valore del patrimonio informativo della pubblica amministrazione mediante l'uso di strumenti di analisi finalizzati alla generazione di nuove conoscenze e alla diffusione di informazioni verso cittadini e imprese;
- migliorare e semplificare l'interoperabilità e lo scambio dei dati tra le pubbliche amministrazioni cercando di evitare molteplici copie degli stessi dati e di consentire un accesso standardizzato a un dato sempre aggiornato;
- incentivare la diffusione degli open data e renderne più efficace la fruizione tramite la centralizzazione e la redistribuzione dei dati pubblici attraverso delle API;
- favorire la gestione e l'analisi esplorativa dei dati al fine di migliorare la conoscenza dei fenomeni descritti da quest'ultimi e sviluppare applicazioni che offrano servizi innovativi ai cittadini, alle imprese e alle pubbliche amministrazioni;
- promuovere iniziative di ricerca scientifica su tematiche di interesse specifico per la pubblica amministrazione, favorendo la collaborazione con università ed enti di ricerca.

Architettura del DAF. Nell'architettura del DAF possiamo distinguere due componenti principali: il dataportal e la piattaforma big data.

Il dataportal rappresenta l'interfaccia utente per l'utilizzo delle funzionalità implementate nel DAF e si compone di:

- il catalogo dei dataset per la gestione sia dei dati contenuti nel DAF sia dei dati provenienti dalle PA;
- di interfacce utente per l'utilizzo dei tool di Data Analysis e Data Visualization;
- di strumenti, quali dashboard, data stories e widget, attraverso i quali è possibile pubblicare le analisi svolte e i risultati ottenuti al fine di collaborare e condividere le nuove conoscenze con altri utenti.

La piattaforma big data è composta da diverse componenti:

- un data lake dove vengono memorizzati i dati di interesse per le pubbliche amministrazioni. Tra questi troviamo i dati generati direttamente dalle pubbliche amministrazioni, i dati generati dai sistemi informativi utilizzati dalle PA (es. file di log) e i dati provenienti dal web e dai social network che possano essere interessanti per gli obiettivi della pubblica amministrazione;
- strumenti di Data Analysis e data visualization che permettono di esplorare i dati disponibili, fare analisi su questi e creare rappresentazioni grafiche come quella riportata nella figura 2.4.

La figure 2.4 riporta una delle diverse visualizzazioni grafiche appartenente alla dashboard "Movimenti turistici in Sardegna dal 2013 al 2016"[14] creata tramite gli strumenti e le interfacce del DAF. L'analisi riportata in figura si pone l'obiettivo di indagare sulla provenienza dei turisti che, nel periodo che va dal 2013 al 2016, si sono recati in Sardegna. Tra i risultati ottenuti è emerso che la Germania rappresenta il paese con il maggior numero di turisti stranieri, seguito dalla Francia e dalla Svizzera; mentre a livello italiano è la Lombardia che si posiziona al primo posto.

2.5 Data Analysis e Informatica Giuridica

Nelle sezioni 2.1 e 2.3 è stato trattato l'ambito della Data Analysis e come questo si sia evoluto negli ultimi decenni. Si è discusso di come il mondo della

[14] `https://dataportal.daf.teamdigitale.it//#/private/userstory/list/cecf4e7f-5636-42d7-9960-2920ccf8356e` (è richiesta la registrazione)

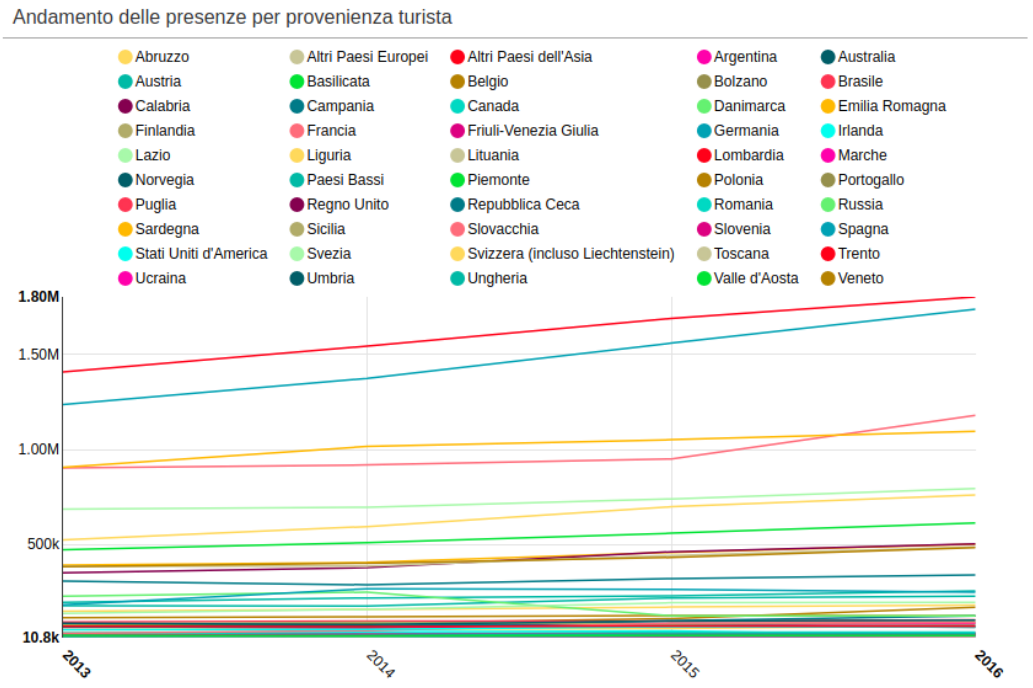

Figura 2.4: DAF: Esempio di Data Visualization

pubblica amministrazione si stia muovendo in direzione degli open data, più precisamente degli Open Government Data, nella sezione 2.4.1. In seguito si è messo in evidenza l'interesse sempre maggiore verso i dati e verso le opportunità che possono nascere dalla loro analisi attraverso la presentazione del Data Journalism (sezione 2.4.2) e del Data & Analytics Framework (sezione 2.4.3).

Alla luce di tale background sul contesto della Data Analysis e sulla rilevanza dei dati digitali, per gli scopi di questa dissertazione si andrà ad esplorare e analizzare il contributo della Data Analysis e della Data Visualization applicata al contesto dell'Informatica Giuridica. Tale contributo ha l'obiettivo di far emergere nuove conoscenze che sono nascoste nelle grandi quantità di documenti legali prodotti nel mondo e che possano essere utilizzate come supporto negli ambiti giudiziari e legislativi. Andando in questa direzione, però, vi è la necessità di descrivere e approfondire diversi argomenti come la trasformazione dei documento legale in formato digitale e i tipi di formato adatti a permettere l'esecuzione di analisi su questi documenti.

Capitolo 3

Informatica Giuridica e strumenti per i documenti legali

In questo capitolo si introdurrà l'Informatica Giuridica per poi trattare la standardizzazione dei documenti legali, passando attraverso la descrizione dei documenti strutturati e dei linguaggi di markup.

3.1 Introduzione all'Informatica Giuridica

L'Informatica Giuridica è la disciplina che tratta l'interazione tra diritto e informatica. Tale disciplina può essere analizzata da due diversi punti di vista poiché gli ambiti di interesse si sviluppano in due direzioni differenti che condividendo lo stesso principio di base: l'interazione tra diritto e informatica. Di seguiro, in accordi con Sartor [Sar16], verrà introdotto questo doppio aspetto che caratterizza l'Informatica Giuridica, i contesti in cui si applica e infine i settori di interesse che la contraddistinguono.

Le due facce dell'Informatica Giuridica, citate precedentemente,sono:

- **Diritto dell'informatica**: con l'utilizzo di nuove tecnologie si vengono a creare mutamenti sociale che a loro volta determinano ulteriori mutamenti sociale e nuovi sviluppi ed usi delle tecnologie. Il diritto dell'informatica si occupa di regolamentare l'introduzione nella società

delle nuove tecnologie e gli effetti da queste prodotte come le nuove esigenze, i nuovi interessi e i nuovi conflitti. Tale area, oltre a dare una risposta alle trasformazioni indotte dalle tecnologie informatiche, contribuisce, a sua volta, a determinare i modi di utilizzo di queste tecnologie, come mostrato nella figura 3.1. Tra i temi chiave trattati dal diritto dell'informatica troviamo la proprietà intellettuale digitale, la protezione dei dati, il commercio elettronico e i reati informatici;

- **Informatica del diritto** (Informatica Giuridica in senso stretto): area che tratta l'utilizzo dell'informatica nei contesti giuridici per l'elaborazione di informazioni giuridiche e il supporto alle attività legali; tale area sarà oggetto di discussione e approfondimento per il resto del capitolo.

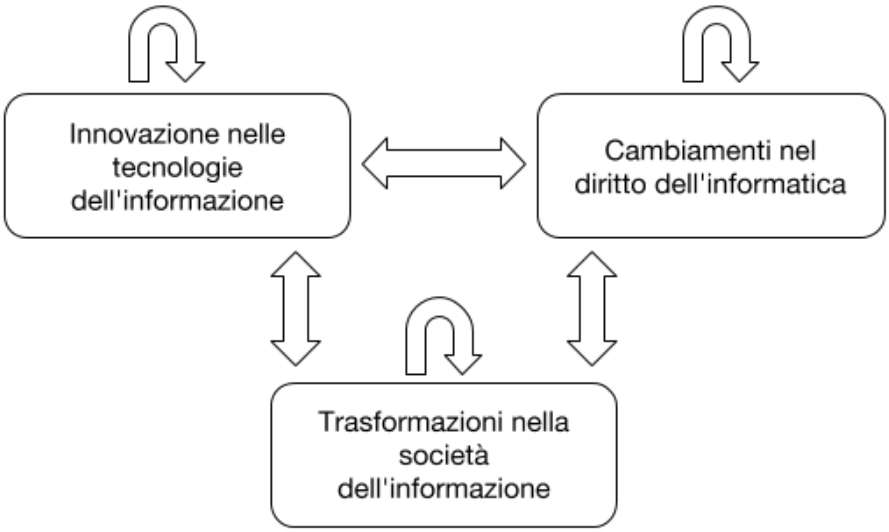

Figura 3.1: Relazione tra diritto, tecnologie e società

Per gli obiettivi di questa dissertazione, da adesso in poi si farà riferimento all'area dell'informatica del diritto.

Cenni Storici. Analizzando la sua storia, l'Informatica Giuridica può essere vista come divisa in fasi che sono fortemente influenzate dal periodo storico e dagli sviluppi tecnologici che lo hanno caratterizzato [BFP08] [Pal15]. I primi esempi di Informatica Giuridica vengono fatti risalire agli anni '50 e '60 quando furono sviluppati i primi database contenenti documenti legali; in questo periodo si notano i primi passi verso l'informatizzazione dei documenti giuridici, come le sentenze, le leggi, etc. Durante gli anni '70 si registra un notevole aumento delle informazioni generate dalle pubbliche amministrazioni e il conseguente sviluppo di molteplici database per la gestione e l'archiviazione dei dati amministrativi. Negli anni '80, conseguentemente alla diffusione dei personal computer, i singoli uffici, legali e amministrativi, iniziano a usufruire delle potenzialità dei computer e degli strumenti software supportati; siamo nell'era dell'automazione degli uffici. Successivamente, negli anni '90 e 2000, con l'avvento di internet e della sua continua espansione, l'Informatica Giuridica inizia ad interessarsi all'aspetto comunicativo tra le organizzazioni legali e il loro pubblico col fine di semplificare e migliorare tale aspetto.

In questa breve introduzione storica si possono già notare alcuni ambiti di competenza dell'Informatica Giuridica e come questi siano emersi in periodi storici differenti. Oggi tra i diversi settori di competenza dell'Informatica Giuridica troviamo:

- **Documentazione giuridica informatica**: realizzazione di banche dati che contengono documenti digitali di dominio legislativo, giuridico e dottrinale, i cui contenuti possono essere selezionati ed estratti automaticamente. Tali raccolte di documenti sono distribuite nella rete e accessibili in modo universale grazie alla struttura dei documenti, la quale segue standard condivisi. In questo settore rientra lo sviluppo di metodi sempre più efficienti e precisi per la ricerca delle informazioni;

- **Sistemi informativi giuridici**: sviluppo di strumenti informatici che possano essere di supporto alle organizzazioni giuridiche per la conservazione, l'estrazione, l'elaborazione e la condivisione delle informazioni. Particolare interesse è rivolto alla sicurezza delle informazioni giuridi-

che, alla correttezza delle elaborazioni svolte su queste ultime e alla loro permanenza nel tempo;

- **Redazione di documenti**: settore che concerne lo sviluppo di software che hanno l'obiettivo di semplificare il lavoro per la creazione di documenti giuridici. A tale fine, vengono messi a disposizione strumenti per la generazione semi-automatica, per la verifica e per la correzione di parte di testo, e per la strutturazione dei testi secondo gli standard che si intende applicare;

- **Apprendimento elettronico del diritto**: settore che riguarda i metodi di utilizzo delle tecnologie informatiche per l'insegnamento del diritto;

- **Modelli informatici del diritto**: si studia e si identificano i modelli che descrivono le strutture rappresentanti conoscenze giuridiche al fine di permette elaborazioni automaticamente su tali conoscenze;

- **Determinazioni giuridiche**: realizzazione di sistemi informatici che applicano i modelli di rappresentazione delle conoscenze giuridiche in modo interattivo o automatico;

- **Deontologia e epistemologia**: settore che concerne, da un lato, lo studio delle condizione in cui l'applicazione degli strumenti informatici esalta e promuove i valori giuridici, dall'altro l'analisi dell'impatto che l'applicazione di questi strumenti crea sulle pratiche del diritto.

I diversi ambiti di competenza dell'Informatica Giuridica, appena descritti, vengono impiegati nelle varie aree applicative che questa disciplina, diventata sempre più ampia e rilevante negli anni, è arrivata ad abbracciare. Le aree in questione sono:

- **Informatica legislativa**: area che concerne l'ambito legislativo. Si occupa dello sviluppo di strumenti informatici per la gestione dei processi legislativi, e di strumenti che mirano a migliorare le singole fasi di tali processi;

- **Informatica giudiziaria**: area che concerne l'ambito giudiziario. Mira allo sviluppo di metodi informatici per la gestione dei processi civili, penali, amministrativi e per una migliore interazione tra le parte in gioco nel processo;
- **Informatica amministrativa**: area che concerne l'ambito della pubblica amministrazione. Studia e sviluppa strumenti informatici che mirano a rendere più efficace e più efficiente la gestione delle procedure amministrative, l'archiviazione e la documentazione dei provvedimenti, l'accesso ai dati pubblici e la comunicazione tra cittadini e pubblica amministrazione;
- **Informatica delle professioni giuridiche**: area che concerne lo studio e lo sviluppo di soluzioni informatiche per il miglioramento delle attività svolte nell'ambito delle professioni giuridiche.

Un settore di grande rilevanza per l'Informatica Giuridica che rientra nell'ambito della documentazione giuridica informatica (descritta precedentemente) e si applica a tutte le aree applicative appena discusse è quello concernente l'integrazione e la standardizzazione dei documenti legali presenti in rete, come le sentenze, gli atti, i documenti amministrativi, etc.

3.1.1 Standardizzazione dei documenti giuridici

L'ambito della standardizzazione dei documenti legali all'interno dell'Informatica Giuridica mira alla creazione di documenti giuridici tramite degli standard condivisi. Tali standard permettono l'arricchimento dei documenti con informazioni semantiche necessarie al fine di creare una base documentale che sia di supporto allo sviluppo di servizi da parte di enti istituzionali e alla semplificazione della cooperazione e all'interoperabilità tra differenti istituzioni, parlamenti e amministrazioni.
A questi obiettivi se ne aggiungono altri che vanno a determinare la direzione intrapresa da questo ambito. Quindi tra gli obiettivi relativi alla standardizzazione dei documenti giuridici abbiamo:

- rendere disponibili e accessibili tutti i documenti legali in rete attraverso un certo formato digitale;

- possibilità di ricostruire automaticamente l'ultima versione di un dato documento giuridico;
- migliorare le funzionalità di ricerca tra i documenti grazie all'utilizzo dei metadati che li arricchiscono;
- creare standard per la struttura, i riferimenti e gli identificatori dei documenti giuridici;
- promuovere l'interoperabilità tra applicazione e sistemi informativi che gestiscono documenti giuridici;
- promuovere servizi e strumenti sia per gli enti che gestiscono la creazione dei documenti sia per i cittadini che ne beneficiano.

Esistono diverse proposte su come i documenti legali, una volta prodotti, possano diventare delle risorse condivise al fine di essere riusate in modo decentralizzato e autonomo. Negli anni si sono delineati due filoni principali:

- **Un sistema centralizzato** dove vi è un database unico che contiene tutti i documenti legali che sono arricchiti tramite uno stesso linguaggio di markup. Tale database prevede un punto di accesso universale e un processo editoriale centralizzato per la sua creazione e il suo aggiornamento. Questo processo ha il compito di controllare i documenti in input, normalizzarli e formattarli in modo uniforme;
- **Un sistema distribuito** che si basa su uno o più punti di accesso ai documenti legali che sono archiviati in modo distribuito tra le singole autorità. Quindi i documenti possono risiedere nel sistema informativo dell'autorità che li ha creati ma essere reperibile da diversi punti di accesso messi a disposizione da organizzazioni legali, pubbliche o private.

Entrambi questi approcci portano con sè vantaggi differenti come, ad esempio: l'affidabilità dei testi legali in quanto forniti dall'unica fonte ufficiale, nel caso dell'approccio centralizzato; l'autonomia delle singole autorità di organizzare e arricchire i documenti legali secondo le proprie necessità, nell'approccio distribuito.

Al fine di prendere i vantaggi da entrambi gli approcci, negli ultimi decenni, si è andato in direzione di una sintesi che si basa sulla produzione decentralizzata di documenti legali strutturati in accordo con uno standard condiviso. Quindi il focus non è più rivolto a chi genera o elabora i documenti, bensì nell'adottare uno standard comune che specifichi come generare degli identificativi univoci per i documenti e come arricchirli attraverso elementi che ne incrementino il valore semantico [BFP08].

Ad oggi esistono diversi standard per i documenti giuridici; alcuni di questi standard sono presentati all'interno di questa dissertazione, nelle sezioni 4.2 e 4.3.

3.2 Markup dei documenti strutturati

Nella sezione precedente si è discusso dell'interesse dell'Informatica Giuridica nella trasformazione dei documenti legali in documenti digitali, arricchiti tramite informazioni semantiche. Al fine di approfondire tale argomento saranno trattati i documenti strutturati come mezzo per la digitalizzazione, e i linguaggi di markup come strumento per la loro creazione.

3.2.1 Documenti strutturati e linguaggi di markup

I documenti strutturati[1] sono dei documenti elettronici che contengono sia il contenuto reale del documento, sia informazioni aggiuntive riguardanti la struttura logica, ovvero informazioni che si riferiscono a come il documento è organizzato, alla sua gerarchia interna e alle relazione esistenti tra le sue parti[2]. Tali informazioni, a secondo dell'obiettivo da raggiungere, possono essere utilizzate in modi differenti come: descrivere la struttura con cui il documento è stato pensato, ad esempio un documento che contiene un titolo, un preambolo, il testo centrale e le conclusioni; inserire informazioni semantiche relative all'intero documento o ad alcune sue parti ad esempio i

[1] *Structured document*, `https://en.wikipedia.org/wiki/Structured_document`

[2] *Logical Structure, Springer Link*, `https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-39940-9_213`

riferimenti a risorse esterne; inserire informazioni riguardanti la presentazione del contenuto.

Queste informazioni possono essere codificate direttamente nel documento insieme al contenuto tramite l'utilizzo dei linguaggi di markup. Tali linguaggi permettono infatti di annotare parti del testo al fine di aggiungere informazioni che portano con sé un significato che sia interpretabile e processabile da un elaboratore.

Esistono diverse tipologie di markup, alcune delle quali sono riportate di seguito [CRD87]:

- **Presentazionale**: markup presente nei tradizionali software di word processing e usato per indicare effetti grafici e non solo, al fine di rendere più chiara la presentazione del contenuto. Si basa sull'inserimento di codice binario nel mezzo del testo ed è solitamente nascosto sia all'utente che all'autore;

- **Procedurale**: il markup è incastrato con il testo del documento e serve ad indicare le istruzioni da eseguire ad un programma che deve processare il documento stesso. Questo tipo di markup è solitamente visibile e direttamente gestito dall'autore del documento;

- **Descrittivo**: il markup descrittivo è usato per arricchire il testo con delle etichette, piuttosto che per specificare istruzione su come deve essere processato. L'obiettivo di questo tipo di markup è di arricchire il testo con informazioni relative alla sua struttura e al significato degli elementi che lo compongono, come ad esempio, il tipo dell'elemento e le relazione tra elementi differenti. Queste informazioni permettono ai documenti di poter essere interpretati e processati da un elaboratore e quindi di poter essere utilizzati in vari contesti come quello della Data Analysis. Due dei principali esempi che rientrano in questa categoria sono HTML e XML.

3.2.2 XML: eXtensible Markup Language

XML è un linguaggio di markup descrittivo sviluppato da W3C[3] come standard aperto che, nel tempo, è stato ampiamente adottato per rappresentare dati e documenti che siano human-readable e machine-readable. L'idea base di XML è quella di permettere annotazioni sul testo tramite la definizione di elementi, chiamati tag, che funzionano come delle etichette che specificano alcune caratteristiche del testo o della sua struttura. Oltre a questo tipo di annotazioni, XML permette l'aggiunta di metadati, ovvero elementi che non caratterizzano direttamente parti di testo ma che aggiungono informazioni semantiche non strettamente collegate alla struttura; ad esempio le tematiche trattate in un documento giuridico sono dei metadati. La principale caratteristica di XML è data dalla sua struttura: gerarchica, rigorosa, estensibile e soprattutto flessibile in quanto ogni comunità che lo utilizza può definire il suo vocabolario di tag, così da definirne un insieme che sia strettamente collegato all'obiettivo che si vuole raggiungere.

Tra le altre interessanti caratteristiche di XML troviamo:

- **open-source**: nessun ente possiede i diritti sul linguaggio, esso è condiviso con la comunità in rete ed è distribuito con una licenza open-source;
- **machine-readable**: i documenti in XML, diversamente da altri formati (es. PDF), possono essere interpretati e processati da un elaboratore che ne può manipolare ogni sua parte;
- **technology-neutral**: XML può essere usato con qualsiasi programma non proprietario quindi non vi sono vincoli relativi agli strumenti utilizzabili.

Tali peculiarità fanno emergere un'ulteriore caratteristica di XML, la quale lo rende un linguaggio interessante per la gestione di documenti giuridici. XML garantisce infatti l'accessibilità e la preservazione a lungo termine dei documenti poiché è un linguaggio open-source e non vincolato da alcuna tecnologia.

[3] *World Wide Web Consortium*, `https://www.w3.org/`

Come accennato precedentemente, al fine di poter utilizzare XML per la generazione di documenti strutturati è necessario definire un vocabolario di tag e di regole associate che ne specifichino le relazioni che intercorrono tra essi. Per la definizione dei tag e delle regole che li governano si possono utilizzare due tipologie di strumenti, i Document Type Definitions (DTD) e gli schemi XML. Entrambi questi strumenti possono essere visti come una guida in cui vengono definite le regole che governano i tag appartenenti ad un certo vocabolario. Una volta che un documento viene generato è possibile verificare la sua correttezza attraverso particolari strumenti che controllano il rispetto delle regole definite nello schema adottato [PV12].

3.2.3 Standard XML come supporto alla Data Analysis

In questo capitolo abbiamo introdotto l'Informatica Giuridica e l'interesse di tale disciplina verso la standardizzazione dei documenti legali. Successivamente sono stati descritti gli strumenti per arrivare a tale obiettivo, ovvero i linguaggi di markup e il linguaggio XML. Questo background, insieme alla descrizione della Data Analysis (capitolo 2), rappresenta la base su cui presentare alcuni esempi relativi alla Data Analysis applicata all'Informatica Giuridica che saranno affrontati nel capitolo successivo.

Capitolo 4

Data Analysis applicata all'Informatica Giuridica

Nel capitolo 2 si è parlato della Data Analysis, della grande rilevanza che ricopre e infine, di come gli enti istituzionali si siano avvicinato a questo settore. Si è introdotto il concetto di Open Government Data, movimento che mira alla divulgazione dei dati provenienti dagli enti istituzionali, il Data Journalism, e il DAF. Il capitolo termina introducendo il lettore all'utilizzo della Data Analysis per l'elaborazione di documenti giuridici forniti in un certo formato digitale.

Nel capitolo 3 è stata presentata l'Informatica Giuridica come disciplina che mira alla creazione di standard per la rappresentazione digitale dei documenti giuridici. In seguito sono stati introdotti i documenti strutturati e il linguaggio XML come gli strumenti necessari per portare i documenti giuridici in formato digitale mantenendo le caratteristiche fondamentali del mondo giuridico come la validità del testo negli anni.

In questo capitolo si cercherà di mettere insieme i due ambiti. Nella sezione 4.1 verranno presentati degli esempi che fanno emergere le potenzialità nell'applicare la Data Analysis all'ambito dell'Informatica Giuridica. Nelle sezione 4.2.2 e 4.3 saranno introdotti alcuni standard XML che rappresentano uno strumento di supporto fondamentale affinchè sia possibile eseguire analisi complesse sui documenti giuridici.

4.1 Esempi di Data Analysis

In questa sezione saranno affrontati tre esempi che mostrano come si può applicare la Data Analysis all'Informatica Giuridica e che tipo di risultati possono essere estrapolati.

Code4Italy. Code4Italy[1] è un progetto sviluppato all'interno del CIRSFID[2] che è incentrato sull'analisi e sulla creazione di rappresentazioni semplici e intuitive di alcuni aspetti significativi del procedimento legislativo del Parlamento italiano. Alcune di queste rappresentazioni sono: grafici comparativi tra le legislature che vanno dalla XIII alla XVII e che mettono in evidenza la natura delle iniziative, i tempi medi di approvazione delle leggi e le proporzioni fra quelle presentate e quelle approvate; visualizzazione del processo legislativo nei due rami del Parlamento; visualizzazione dei tempi medi di approvazione di un progetto di legge suddiviso per tipologia (conversione, bilancio, ordinaria).

Per questo progetto sono stati utilizzati i dataset forniti dal Senato e dalla Camera, interrogando tramite query SPARQL i loro endpoint. Successivamente i dataset sono stati esportati tramite programmi Javascript in JSON o CSV e, tramite Python, quest'ultimi sono stati elaborati per ottenere le visualizzazioni; tali visualizzazioni sono state create tramite la librerie d3js[3] e le API fornite da Google Visualization[4]. La figura 4.1 mostra una delle analisi sviluppate all'interno del progetto Code4Italy dove vi vuole mettere in evidenza le proporzioni fra le varie tipologie di iniziativa. Ogni esagono rappresenta i diversi contesti di partenza dell'iniziativa (es. parlamentare, governativa); all'interno degli esagoni, ogni cerchio rappresenta un progetto di legge e può essere caratterizzato da diversi colori: rosso se ha concluso negativa il suo iter; verde se è stato approvazione in legge; bianco o giallo se è ancora sotto esame presso uno dei due rami del Parlamento o se la sua discussione è in programma. Osservando l'esagono relativo alle iniziative governative si possono notare i cerchi bordati in blu, che rappresentano i decreti

[1] *Code4Italy*, `http://code4italy.cirsfid.unibo.it/`
[2] `http://www.cirsfid.unibo.it/`
[3] *Data-Driven Documents*, `https://d3js.org/`
[4] *Google Charts*, `https://developers.google.com/chart/`

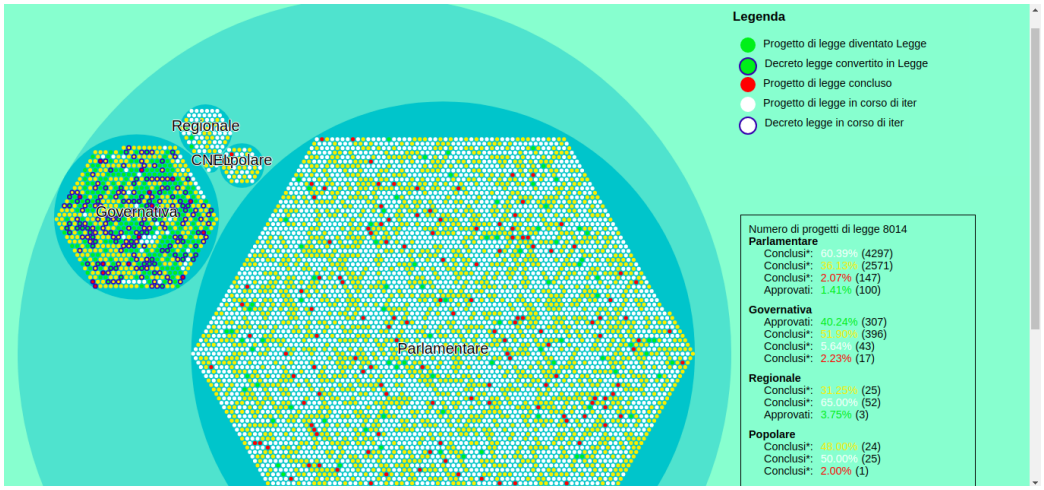

Figura 4.1: Visualizzazione realizzata nel progetto Code4Italy

legge. Da questa analisi è emerso che vi sono moltissime iniziative parlamentare che non trovano spazio di dibattito all'interno degli organi legislativi e soprattutto, solo una percentuale molto bassa dei progetti di legge giunge a un esito positivo. Viceversa le iniziative governative giungono a esito positivo con una buona percentuale.

Network of French legal codes. Nel lavoro di Mazzega [MBB09] viene proposta un'analisi delle leggi francesi. Sono stati selezionati 52 codici legali, sulla base di particolari criteri, che hanno permesso lo sviluppo di rete di connessioni. Quest'ultima rappresenta i codici come vertici e le citazione presenti tra i codici come archi. Il risultato finale ha fatto emergere l'esistenza di un gruppo di dieci codici che citano maggiormente e che sono maggiormente citati. Questo insieme è caratterizzato da una fitta rete di connessioni, formando una sorta di "club esclusivo". La figura 4.2 è la rappresentazione della rete sviluppata in questo progetto dove i 52 codici sono rappresentati dai vertici, e le loro connessioni dagli archi.

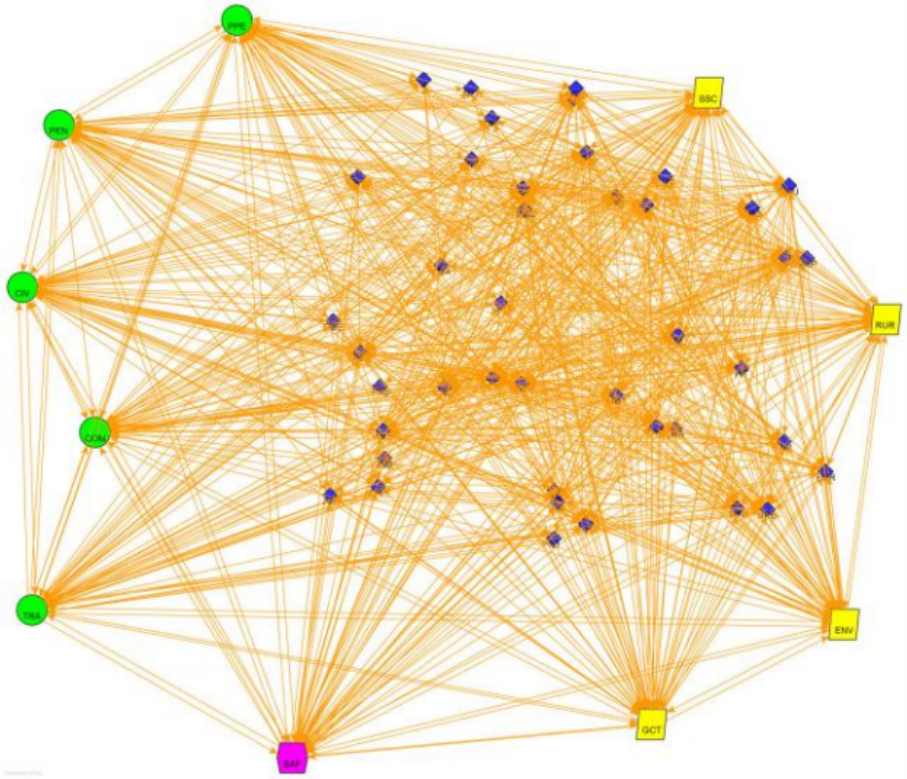

Figura 4.2: Grafo associato alla rete dei codici legali francesi

Socievole. Socievole[5] (SOCIal Evolution and Visualization Of Legal acts) è un altro progetto sviluppato all'interno del CIRSFID che utilizza i documenti della legislazione inglese, in formato *Akoma Ntoso* (standard XML descritto nella sezione 4.3), disponibili nel portale legislation.gov.uk[6]. Questi documenti XML, una volta archiviati in un database eXist[7], sono elaborati tramite tecniche di NLP per rilevare i titoli e le keyword che serviranno all'estrazione dei temi sociali. Quest'ultimi sono alla base della timeline che è stata realizzata come strumento per rappresentare l'evoluzione dei temi sociali, dal 1800 al 2016, attraverso la legislazione. La figura 4.3 mostra una parte della timeline con i vari temi che sono stati estratti per gli anni visualizzati.

[5] *Socievole*, `http://sinatra.cirsfid.unibo.it/socievole/home`
[6] *legislation.gov.uk*, `http://www.legislation.gov.uk./`
[7] *eXist-db*, `http://exist-db.org/exist/apps/homepage/index.html`

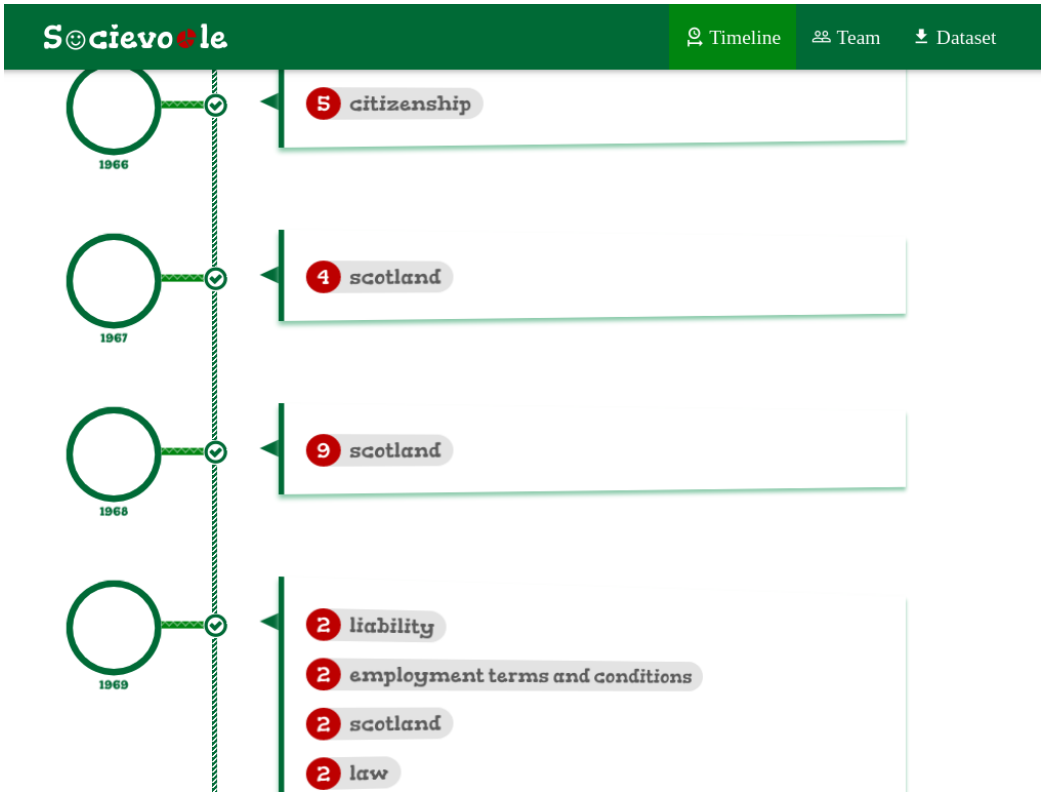

Figura 4.3: Timeline realizzata nel progetto Socievole

I tre esempi riportati evidenziano l'interesse verso l'analisi dei documenti giuridici. Per arrivare a questo scopo è necessario che tali documenti siano strutturati in modo da poter essere processati tramite un elaboratore. Inoltre, per eseguire analisi sempre più complesse e non puramente testuali è necessario che i documenti presentino informazioni semantiche relative alla struttura e al significato delle parti che lo compongono. Gli strumenti che ci consentono di raggiungere tali obiettivi sono gli standard XML poiché permettendo una serie di arricchimenti del testo e soprattutto garantiscono un insieme di proprietà che sono necessarie quando si tratta di documenti giuridici.

4.2 Standard XML per i documenti legali

Come discusso nella sezione 3.1.1, all'interno dell'Informatica Giuridica c'è molto interesse verso lo sviluppo di standard condivisi per i documenti legali al fine di beneficiare delle opportunità che nascono andando in questa direzione. Nella sezione 3.2.1, invece, si è visto come il linguaggi XML può essere adottato per la creazione di documenti strutturati e di come le sue caratteristiche lo rendano adatto al mondo dell'Informatica Giuridica.

Il linguaggio XML insieme agli schemi (DTD o XML) che lo caratterizzano e ad altri strumenti di supporto come i meccanismi per la generazione di identificativi univoci per le risorse, sono alla base degli standard XML. E' importante sottolineare come non tutti gli standard XML sono appropriati per le esigenze del mondo giuridico, infatti possono essere individuate alcune caratteristiche e alcune componenti che contraddistinguono uno standard utilizzabile, robusto e duraturo [PV12], alcune di queste sono:

- l'interoperabilità tra le diversi istituzioni e le diverse risorse;
- la compatibilità con altri standard W3C[8] al fine di garantire stabilità con il passare degli anni;
- supporto alle redazioni delle legislature di differenti nazioni;
- preservare la validità dei documenti legali per lunghi periodi; questo argomento è stato già discusso in precedenza dove è emerso che XML è un linguaggio appropriato per soddisfare questo requisito;
- strumenti di supporto per la gestione degli identificatori per le risorse all'interno del web; questi devono essere espressivi, invarianti e persistenti nel tempo così da garantire dei riferimenti a risorse legali che siano stabili e invariati nel tempo;
- separazione tra il contenuto prodotto dalle autorità e quindi ufficiale, e i metadati che provengono da terze parti come uffici governativi o editori locali.

[8] *World Wide Web Consortium*, `https://www.w3.org/`

4.2.1 Strumenti per gli standard XML

In precedenza si è accennato al fatto che gli standard per i documenti giuridici non sono formati solamente dagli schemi XML bensì sono arricchiti da diversi strumenti che hanno lo scopo di supportarlo e di permettere la sua affermazione. Di seguito verranno introdotti alcuni di questi strumenti [BFP08].

Convertitore . Il convertitore è uno dei principali strumenti per aumentare la diffusione di uno standard per i documenti legali. Tale strumento ha due obiettivi ovvero quello di convertire i documenti attualmente prodotti in modo tradizionale, nello standard XML; e quello di convertire nello standard XML tutti i documenti già esistenti come ad esempio le leggi già approvate. Questo lavoro ha una grande importanza poiché permette l'utilizzo in modo completo dei riferimenti tra i vari documenti legali, datati e recenti che siano. I convertitori si basano sull'idea di una conversione semi automatica, ovvero dove vi è un processo automatico per individuare nel miglior modo possibile la struttura del documento, e un processo manuale per la verifica e la conferma del risultato ottenuto dal processo automatico. Ovviamente, migliore è il convertitore e meno lavora sarà richiesta all'utente che si occuperà della verifica e del completamento delle parti mancanti.

Editor . Un'altro strumento fondamentale per la generazione di documenti giuridici in uno standard XML è l'editor. Possiamo distinguere tre tipologie differenti di utilizzo degli editor:

- editor come interfaccia per il controllo e la verifica del processo automatico svolto dal convertitore (descritto precedentemente). In questo scenario quindi, viene utilizzato l'editor al fine di modificare o aggiungere parti del documento che la conversione ha trattato in modo errato o che sono state completamente saltate;

- editor come strumento per il markup manuale di un documento legale fornito in un determinato formato. In questo caso, l'editor è fornito con tutti gli strumenti adatti a modificare o aggiungere qualsiasi tipo

di markup, oltre che a verificare la validità del documento secondo lo schema utilizzato;

- editor come strumento per la creazione di documenti legali partendo dal documento vuoto. In questo caso si ha la possibilità di inserire sia parti di testo che elementi di markup.

Name resolvers . I documenti digitali sono sempre più frequentemente archiviati nella rete e quindi accessibili specificando il loro indirizzo. Tali indirizzi sono fondamentali perché determinano in modo univoco i vari documenti e permettono di creare dei riferimenti tra loro. Un elemento essenziale per uno standard XML è la naming convention ovvero un meccanismo per la creazione di identificatori che permettono di riferirsi e accedere alle risorse in rete restando indipendenti dall'architettura con cui queste sono state archiviate e gestite. La naming convention deve essere combinata con un name resolver ovvero uno strumento che, dato l'identificativo di una risorsa che non è connesso ad alcuna architettura, riesce a fornire l'indirizzo reale della risorsa, dipendente dall'architettura sottostante, per poter accedere e utilizzarla.

Strumenti di validazione . Gli strumenti di validazione sono essenziali per verificare che il documento creato rispetti tutti i requisiti. Tali strumenti dovranno verificare che: la struttura e il contenuto rispettino i vincoli definiti dagli schemi XML; tutti i riferimenti presenti nel documento siano correttamente referenziati; tutti i metadati presenti nel documento siano corretti e completi.

4.2.2 Classificazione degli standard XML

Oggi esistono una moltitudine di standard XML che si applicano al mondo giuridico, ognuno con le proprie peculiarità e i propri limiti. In accordo con il lavoro della Palmirani e Vitali [PV12], tali standard possono essere classificati in quattro generazioni basandosi sulle loro caratteristiche principali:

- **Prima Generazione**: questa generazione si focalizza sulla descrizione del testo e della struttura con un approccio simile al modello dei database o al modello dei word processing;

- **Seconda Generazione**: in questa generazione vi è più interesse verso la modellazione e la descrizione del testo, della struttura e dei metadati. E' caratterizzata da lunghe liste di tag, complesse inclusioni definite nei DTD o negli schemi XML e molte definizione sovrapposte relative al testo e ai metadati;

- **Terza Generazione**: tale generazione differentemente dalle precedenti, si basa sui pattern. I pattern definiscono le proprietà di una classe e la sua grammatica fornendo un content model e specificando il comportamento e la gerarchia della classe in relazioni alle altre classi. Questa generazione si fonda inoltre sulla chiara divisione tra testo, struttura, metadati e ontologie. Con l'utilizzo dei pattern si assicura la chiarezza del design grazie alla definizione di regole generali e non più di vincoli diretti sul markup, ma si perdono i vincoli prescrittivi;

- **Quarta Generazione**: quest'ultima generazione fa utilizzo dei pattern accompagnandoli con dei vincoli grammaticali al fine di risolvere il problema, presente nella terza generazione, della mancanza di vincoli prescrittivi.

Tra gli standard che vedremo in questa dissertazione troviamo *Akoma Ntoso*, appartenente alla terza generazione, e lo standard NIR (NormeInRete), appartenente alla seconda generazione.

NormeInRete (NIR) . NormeInRete è un progetto italiano iniziato nel 1999 che porta con se i seguenti risultati: un portale web che fornisce un punto di accesso unico per le ricerche sul corpo legislativo italiano; uno standard XML per la rappresentazione di documenti legali; uno standard per l'identificazione persistente dei documenti legali [BFP08]. Lo standard per l'identificazione dei documenti legali è definito in accordo con gli URN (Uniform Resource Name) al fine di associare ogni documento legale a un identificativo univoco, in un formato standard, indipendente dalla sua reperibilità

in rete e dalla posizione fisica dove è allocato. Questo permette di esprimere riferimenti in modo stabile e indipendente dal luogo dell'archiviazione.

La gestione dei link nella rete non è facilmente scalabili per gli obiettivi di uno standard per i documenti legali poiché essi si basano sulla locazione fisica espressa tramite gli URL e quindi presentano problematiche come, ad esempio, la perdita di validità nel tempo della locazione e l'impossibilità di riferirsi a una risorsa non ancora disponibile in rete. Al fine di superare tale problema, NIR ricorre a un sistema di riferimenti che si basa sull'assegnazione di identificatori uniformi per ogni risorsa legale e l'utilizzo di RDS (Resolver Discovery Service), metodo di risoluzione che è in grado di recuperare l'oggetto corrispondente ad un certo identificativo. Segue l'esempio di un identificatore utilizzato nello standard NIR.

```
Decreto del Ministero di Firenze del 20-12-99
urn:nir:ministero.finanze:decreto:1999-12-20;nir-3
```

Una peculiarità di NIR è il supporto parallelo ai documenti che seguono le regole di redazione standard e ai documenti, scritti precedentemente, che non seguono tali regole [MMS02]. Lo standard XML per la rappresentazione di documenti legali di NormeInRete rispecchia tale peculiarità ed è costituito da tre DTD:

- **DTD flessibile**: DTD che presenta poche regole ed è usato per i documento che non seguono le regole di redazione, come quelli generati precedentemente allo standard;
- **DTD base**: rappresenta un sottoinsieme del DTD completo ed è usato per allenare gli utenti che stanno per adottare tale standard;
- **DTD completo**: DTD contenente tutti i tag e tutte le regole previste dallo standard; è usato per la generazione di nuovi documenti legali.

Al fine di supportare la creazione di documenti legali che siano in accordo con lo standard NIR è stato sviluppato un editor specializzato. L'editor di NIR nasce quindi per lavorare con lo standard URN e i DTD standard di NIR e mira a fornire un supporto sia per la redazione di nuovi documenti giuridici, sia per lavorare con documenti già esistenti [BFS03].

4.3 Akoma Ntoso: verso uno standard universale

Akoma Ntoso[9] (Architecture for Knowledge-Oriented Management of African Normative Texts using Open Standards and Ontologies) è uno standard XML sviluppato nell'ambito di un progetto sponsorizzato da UN DESA[10] (United Nations Department for Economic and Social Affairs), con l'obiettivo di migliorare la qualità dei servizi parlamentari, facilitarne le attività svolte all'interno, e promuovere la partecipazione cittadina a tali processi. Oggi, *Akoma Ntoso* è uno standard OASIS sotto il nome di *LegalDocML*.

Akoma Ntoso permette la definizione di rappresentazioni XML tecnologicamente neutrali e machine-readable di documenti parlamentari, legislativi e giudiziari al fine di supportare la creazione di nuovi servizi e di migliorare la cooperazione tra diversi enti in un contesto mondiale. Quest'ultima caratteristica lo differenzia da molti altri standard XML che invece sono pensati e sviluppati per l'utilizzo all'interno di un certo contesto o di una certa nazione, come il caso di NIR (sezione 4.2.2) che è ristretto al contesto italiano.

L'orientamento verso uno standard universale e non pensato per un singolo contesto, apre la strada a molteplici vantaggi come ad esempio nell'ambito legislativo dove può promuovere la collaborazione tra parlamenti di diverse nazioni al fine di favorisce lo scambio di conoscenze e la possibilità di scoprire e imitare certe attività che si reputano positive anche per il proprio paese [BFP08].

Il logo *Akoma Ntoso*, mostrato nella figura 4.4, è un simbolo utilizzato dagli Akan, popolo dell'Africa Occidentale, per rappresentare intesa e accordo [UN18].

4.3.1 Tre livelli: testo, struttura e metadati

I documenti *Akoma Ntoso* sono caratterizzati da una struttura a tre livelli:

- **livello testuale**: fornisce una rappresentazione XML del contenuto originale del documento legale;

[9] *Akoma Ntoso*, `http://www.akomantoso.org/`
[10] *UN DESA*, `https://www.un.org/development/desa/en/`

Figura 4.4: Akoma Ntoso

- **livello strutturale**: fornisce un'organizzazione gerarchica delle parti presenti nel livello testuale assegnando un ruolo ai blocchi e ai frammenti presenti nel testo in modo tale da specificare, ad esempio, articoli, clausole, blocchi semplici, etc.;
- **livello dei metadati**: insieme di conoscenze legali che possono essere aggiunte al documento come interpretazioni soggettive del contenuto testuale.

Ogni documento presenta esattamente un livello testuale e un livello strutturale. Invece è possibile avere più livelli dei metadati poiché *Akoma Ntoso* gestisce l'archiviazione di differenti interpretazioni per uno stesso documento in modo tale da poterle utilizzare alternativamente, o in maniera cumulativa o anche, in maniera comparativo.

Il livello relativo alla struttura è caratterizzato da una netta separazione tra il contenuto, ovvero il testo originale fornito da un ente autorevole, e i metadati, ovvero le informazioni aggiunte da terzi che sono derivate da interpretazioni soggettive. Questa separazione mira a preservare e garantire l'autenticità e la validità del documento legale nel tempo.

Akoma Ntoso non impone l'utilizzo di alcuna ontologia, questa caratteristica mira a favorire l'indipendenza da qualsiasi tecnologia e quindi, a garantire uno standard robusto e duraturo. Lo standard *Akoma Ntoso* infatti definisce implicitamente una struttura ontologica per la rappresentazione dei metadati che si fonda sulle Top Level Classes (TLC), classi generiche di istanze come ad esempio: Person, Role, Concept, etc. Tale struttura è definita implicitamente poiché non vi è alcuna implementazione esplicita e condivisa dell'ontologia che definisce queste classi e le loro relazioni [BCP10]. L'utilizzo delle Top Level Classes ha due obiettivi principali: fornire un insie-

me di classi per l'identificazione delle entità presenti nel documento in modo uniforme ed espressivo; fornire una strutturazione delle proprietà e delle relazioni esistenti con le altre istanze della stessa classe o di classi differenti. Tutte le entità presenti nei documenti *Akoma Ntoso* possono essere associate a determinate classi TLC specificando un certo IRI, il quale dovrà seguire le regole della naming convention, discussa di seguito [OASIS17].

4.3.2 Cenni sullo schema Akoma Ntoso

Akoma Ntoso utilizza due tipologie di schema:

- **Schema Generale**: definisce gli elementi e gli attributi utilizzabili, e un insieme minimale di vincoli che tutti i documenti *Akoma Ntoso* devono rispettare;
- **Schemi Custom**: insieme aperto di schemi che si differenziano per i vincoli definiti, partendo dallo stesso vocabolario di elementi e attributi. Tutti gli schemi custom sono vincolati dallo schema generale, ovvero tutti i documenti che soddisfano uno schema custom devono soddisfare lo schema generale.

Nella sezione 4.2.2 sono state analizzate le diverse generazione di standard XML per il documenti legali [PV12] osservando che lo standard *Akoma Ntoso* si posizione nella terza generazione. *Akoma Ntoso* è caratterizzato quindi dai pattern che definiscono le proprietà di una classe e la sua grammatica fornendo un content model e specificando il comportamento e la gerarchia della classe in relazioni alle altre classi.

Lo schema generale di *Akoma Ntoso* si basa sui sei pattern [UN18] che definiscono i vari content model per gli elementi dello standard. Tali pattern possono essere così riassunti:

- **Marker**: sono elementi senza contenuto che acquisiscono un significato combinando la loro posizione nel testo, il loro nome e i loro attributi. Esistono due principali famiglie di marker: placeholder che possono comparire in qualsiasi posto nel testo; elementi per i metadati che possono essere presenti solamente nella sezione dedicata ai metadati (es. noteRef);

- **Inline**: contenitore di testo che, posizionato insieme a un elemento "block", indica certe caratteristiche semantiche o strutturali (es. person);

- **Block**: indica dei contenitori di testo autonomi da altri elementi, e che possono contenere elementi "inline"(es. paragraph);

- **Container**: l'elemento "container" è formato da un insieme di elementi caratterizzati da tipi differenti (es. act);

- **Hierarchical Container (hcontainer)**: gerarchia di container posti a livelli diversi di annidamento e caratterizzati da un titolo e da una numerazione. Ogni livello può contenere sia altri elementi dello stesso tipo che elementi di tipo "block". Non è permesso usare del testo all'interno della gerarchia ma solamente all'interno dell'elemento "block" appropriato (es. body);

- **Popup**: elementi che, all'interno di un flusso di testo e di elementi "inline", creano strutture indipendenti che non interagiscono con il testo e con gli elementi "inline" che li circondano (es. quotedStructure).

4.3.3 Naming Convention e il modello FRBR

La naming convention di *Akoma Ntoso* identifica, in modo univoco, tutti i concetti e le risorse *Akoma Ntoso* nel web. Di seguito sono riportati i tre principi base della naming convention [OASIS17]:

- **Significatività**: il nome è una descrizione logica e significativa della risorsa, non della sua locazione;

- **Permanenza**: il nome deve rimanere permanente e stabile nel tempo;

- **Invarianza**: il nome deve essere generato a partire da proprietà invariabili della risorsa così che, con una probabilità alta, si ottenga lo stesso nome per la stessa risorse, indipendentemente dal processo, dallo strumento e dalla persona.

Internationalized Resource Identifier (IRI) . Alla base della naming convention di *Akoma Ntoso* vi è il concetto di IRI[11] (Internationalized Resource Identifier) che, diversamente dagli URI, i quali sono limitati a un sottoinsieme di caratteri ASCII, possono contenere tutti i caratteri dello standard UCS [W3C18]. Nei documenti *Akoma Ntoso* vengono utilizzati solamente gli IRI espressi in forma relativa ovvero senza indicare il protocollo e il nome del dominio.

Un'altra caratteristica nell'utilizzo degli IRI, in *Akoma Ntoso*, è la differenza tra un IRI globale e un IRI locale: l'IRI globale è un IRI relativo che inizia sempre con uno slash per indicare che tutte le restanti parti sono specificate; l'IRI locale invece può avere una o più parti mancanti ed è possibile risalire all'IRI globale utilizzando le informazioni reperibili nel documento base.

La naming convention è una parte molto rilevante dello standard *Akoma Ntoso* in quanto fornisce un supporto per la generazione degli identificatori relativi ai concetti e alle risorse. La naming convention gestisce i riferimenti tra risorse differenti, i riferimenti locali al documento e le ontologie implicite (sezione 4.3).

Akoma Ntoso non presuppone che ogni documento sia archiviato tramite un XML *Akoma Ntoso* ma solamente che esista una mapping tra l'IRI che lo identifica e l'URL di un file memorizzato da qualche parte sul web, con il quale è possibile risolvere il riferimento.

Per identificare le risorse e quindi per una gestione efficace dei riferimenti a documenti esterni, sono utilizzati gli URI globali. Invece, per quanto riguardo i riferimenti alle parti interne dei documenti, come componenti, porzioni e frammenti, vengono utilizzati gli IRI locali [OASIS17].

Modello FRBR per l'identificazione dei documenti . *Akoma Ntoso* interpreta i documenti come risorse bibliografiche basandosi sul modello concettuale FRBR (Functional Requirements for Bibliographic Records); tale modello mette in relazione le attività dell'utente relative al recupero e all'ac-

[11] `https://www.w3.org/International/O-URL-and-ident.html`

cesso dei cataloghi online delle biblioteche e dei database bibliografici. Tale modello prevede l'esistenza delle seguenti entità [IFLA98]:

- **Work**: entità che definisce il collegamento tra un documento e il suo ideatore a livello astratto (proprietà intellettuale);
- **Expression**: entità che definisce le varie versioni con cui un documento astratto (work) è espresso, interpretato o rappresentato; le expression possono differire tra loro per la lingua, la versione, etc.;
- **Manifestation**: entità che definisce le diverse manifestazioni di una stessa expression. Ad esempio, due documenti rappresentanti la stessa versione di un work ma che si differenziano per il formato dati utilizzato;
- **Item**: entità che rappresenta le varie copie, possedute concretamente, di una certa manifestation.

Applicando tale modello concettuale al mondo giuridico, ad esempio, si avrà che un documento *Akoma Ntoso* rappresentante una legge individuerà come work il nome della legge rappresentata (es. atto 3 del 2015); come expression il nome della legge in una sua certa versione che può differire per data, lingua, etc. (es. atto 3 del 2015, nella versione che segue gli emendamenti entrati in vigore il 3 luglio 2016); come manifestation il file, in un certo formato (es. XML) che rappresenta l'expression.

A partire da questo modello concettuale, la naming convention di *Akoma Ntoso* definisce come generare gli identificatori univoci per le varie entità (work, expression, manifestation) che caratterizzano ogni documento in questo standard.

La struttura dell'entità work è così composta, definendo per ognuno dei seguenti step, il carattere "/" come separatore [OASIS17]:

- il prefisso "/akn";
- il paese (due caratteri) che deve corrispondere con il contenuto dell'elemento "FRBRcountry" presente nei metadati;
- il tipo del documento (es. act, bill, or debateReport);

- specifica del sottotipo del documento che deve corrispondere con il contenuto dell'elemento "FRBRsubtype" presente nei metadati (opzionale);
- l'ente emanante del documento che deve corrispondere con il contenuto dell'elemento "FRBRauthor" all'interno di "FRBRWork", presente nei metadati (opzionale);
- la data di creazione del documento originale nel formato YYYY-MM-DD oppure YYYY (se sufficiente per lo scopo identificativo), che deve corrispondere con il contenuto dell'elemento "FRBRdate" all'interno di "FRBRWork" presente nei metadati;
- il numero o il titolo o un'altra caratteristica distintiva del work che deve corrispondere con il contenuto dell'elemento "FRBRnumber" o "FRBRname" presente nei metadati (necessario se c'è necessità di disambiguare, sennò opzionale);
- specifica della componente e del frammento (opzionale).

A partire dall'entità work, si costruisce l'entità expression, come segue:

- IRI del work costruito come descritto in precedenza;
- il carattere "/";
- la lingua con cui l'expression è stato redatto (tre caratteri) che deve corrispondere con il contenuto dell'elemento "FRBRlanguage" presente nei metadati;
- il carattere "@";
- la data della versione dell'expression in formato YYYY-MM-DD che deve corrispondere con il contenuto dell'elemento "FRBRdate" all'interno di "FRBRExpression" presente nei metadati; se non si tratta di una legge approvata allora è necessaria la data di presentazione del documento o il numero di versione oppure un'informazione qualsiasi che aiuti nell'identificazione della versione;

- un'ulteriore informazione riguardante la data che specifica il contenuto (opzionale);
- informazione riguardante il creatore del testo che deve corrispondere con il contenuto dell'elemento "FRBRauthor" all'interno di "FRBRExpression" presente nei metadati (opzionale).

Lo stesso meccanismo si applica per determinare l'entità manifestation a partire dall'entità expression:

- IRI dell'expressione definita in precedenza;
- identificativo del creatore del markup che deve corrispondere con il contenuto dell'elemento "FRBRauthor" all'interno di "FRBRManifestation" presente nei metadati (opzionale);
- data del markup che deve corrispondere con il contenuto dell'elemento "FRBRdate" all'interno di "FRBRManifestation" presente nei metadati (opzionale);
- qualsiasi informazione aggiuntiva relativa al markup (opzionale);
- il carattere ".";
- estensione del formato usate per tale manifestation che deve corrispondere con il contenuto dell'elemento "FRBRformat" all'interno di "FRBRManifestation" presente nei metadati.

Per quanto riguarda l'entità item, *Akoma Ntoso* non fa nessuno assunzione riguardante il meccanismo di archiviazione fisico della manifestation e quindi non vi sono regole che governano la struttura degli IRI relativi alle entità item.

Al fine di chiarificare i passi appena descritti, necessari per l'identificazione univoca dei documenti, di seguito è mostrato un parte di un documento XML *Akoma Ntoso*. Tale frammento XML mostra il contenuto dell'elemento "identification", presente all'interno dell'elemento "meta"(elemento per i metadati), in cui vengono descritte le varie entità FRBR insieme ad altre informazioni come ad esempio: FRBRauthor, FRBRdate, FRBRlanguage, etc.

```
<FRBRWork>
        <FRBRthis value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13/!main"/>
        <FRBRuri value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13"/>
        <FRBRalias value="ECLI:EU:C:2014:1758" name="ECLI"/>
        <FRBRdate date="2014-06-12" name="generation"/>
        <FRBRauthor href="#CIRSFID" as="#author"/>
        <FRBRcountry value="eu"/>
        <FRBRsubtype value="summary"/>
        <FRBRnumber value="C-39-13"/>
        <FRBRprescriptive value="false"/>
    <FRBRauthoritative value="false"/>
</FRBRWork>
<FRBRExpression>
        <FRBRthis value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13/ita@/!main"/>
        <FRBRuri value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13/ita@"/>
        <FRBRdate date="2014-06-12" name="markup"/>
        <FRBRauthor href="#CIRSFID" as="#editor"/>
        <FRBRlanguage language="ita"/>
</FRBRExpression>
<FRBRManifestation>
        <FRBRthis value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13/ita@/!main.xml"/>
        <FRBRuri value="/akn/eu/jugment/summary/unibo
            /2014-06-12/C-39-13/ita@.akn"/>
        <FRBRdate date="2018-09-03" name="publication"/>
        <FRBRauthor href="#CIRSFID" as="#editor"/>
        <FRBRformat value="xml"/>
</FRBRManifestation>
```

4.4 Utilizzo dello standard Akoma Ntoso per la Data Analysis

Gli esempi di Data Analysis applicata all'Informatica Giuridica, nella sezione 4.1, hanno mostrato che è possibile estrarre informazioni nuove e interessanti da grandi quantità di documenti strutturati. In seguito si è visto come la strutturazione di questi documenti si basa su standard XML che ne permettono l'arricchimento semantico grazie agli elementi di markup. Nella sezione 4.3 è stato introdotto uno standard XML che si contraddistingue dagli altri standard per il suo carattere universale, la sua flessibilità e la sua ricchezza semantica.

Alla luce di queste informazioni, nel prossimo capitolo, sarà presentato un esempio di progetto inerente all'ambito del Data Journalism, in cui ho sviluppato delle analisi su documenti *Akoma Ntoso* e ho implementato uno strumento per la rappresentazione grafica dei risultati ottenuti. La trattazione di queste analisi e delle rappresentazione grafiche associate è un passaggio fondamentale per la costruzione di un percorso che porti alla verifica della tesi secondo cui la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

Capitolo 5

SOFIA: Structured Open government data For Interactive Analysis

Nel capitolo 2 è stato trattato l'ambito della Data Analysis, la storia, gli obiettivi e la sua rilevanza nel mondo di oggi attraverso la presentazione del concetto di Open Government Data, del DAF e del Data Journalism. Infine si è accennato alle potenzialità che vi sono con l'applicazione della Data Analysis all'Informatica Giuridica, ovvero all'analisi dei documenti giuridici che sempre più frequentemente sono resi disponibili e accessibili in modo digitale.

Nel capitolo 3 si è parlato dell'Informatica Giuridica e di come questa disciplina si occupi della digitalizzazione dei documenti legali attraverso l'utilizzo di standard XML, trattati nel capitolo 4. Quest'ultimi mirano a garantire certe proprietà ai documenti giuridici (sezione 3.1.1) e permettono a un elaboratore di poter processare tali documento al fine di analisi o di ricerche. Questi standard sono uno strumento fondamentale per poter mettere insieme i due ambiti, infomatica giuridica e Data Analysis, con l'obiettivo di beneficiare degli strumenti di Data Analysis nell'ambito giuridico attraverso l'utilizzo di documenti strutturati e machine-readable.

In questo capitolo vedremo lo sviluppo concettuale di *SOFIA*, acronimo di Structured Open government data For Interactive Analysis, che identifica la

dashboard che ho sviluppato e rappresenta una parte del mio progetto di tesi insieme ad *akomando-db*, discusso successivamente. *SOFIA* è uno strumento che permette la fruizione dei risultati ottenuti dalle analisi, fatte su collezioni di documenti legislativi *Akoma Ntoso*, tramite rappresentazioni grafiche. Durante questo capitolo saranno individuate le componenti necessarie allo sviluppo di tale strumento, introducendo quelle già esistenti e accennando a quelle attualmente non disponibili.

5.1 Obiettivi

Per raggiungere gli scopi formulati nella tesi di questa dissertazione ho deciso di definire e implementare delle analisi su documenti *Akoma Ntoso* per poi sviluppare uno strumento per la visualizzazione dei risultati ottenuti, un caso d'uso tipico nel mondo del Data Journalism. L'obiettivo finale delle visualizzazioni è quello di mostrare se e come l'organizzazione strutturale dei documenti del sistema legislativo italiano sia mutata negli anni. Tali visualizzazioni possono aiutare gli uffici legislativi a semplificare le norme[1], rendere più comprensibile e fluido il discorso giuridico[2] e poterlo così governare nell'evolvere del tempo mediante modifiche giuridiche coerenti [PBC18]. Quindi si proverà ad analizzare:

- il numero di articoli presenti nel corpo legislativo, raggruppati per i diversi anni e per le diverse legislature;
- il numero di paragrafi presenti nel corpo legislativo, raggruppati per i diversi anni e per le diverse legislature;
- il numero di articoli e di paragrafi dei documenti legislativi avendo il focus sulla tipologia di documento: decreto legge, decreto legislativo, legge.

Al fine di avere un quadro completo delle analisi eseguite e riuscire a individuare dei trend interessanti, i risultati sono associati all'analisi del numero

[1] `https://ec.europa.eu/info/law/law-making-process/planning-and-proposing-law/better-regulation-why-and-how_it`

[2] `http://leg16.camera.it/temiap/temi16/La_buona_scrittura_delle_leggi.pdf`

medio di caratteri che compongono la parte centrale dei documenti legislativi, quest'ultimi raggruppati in modo coerente con l'analisi associata.

Perché Akoma Ntoso . Si è deciso di utilizzare documenti nello standard *Akoma Ntoso* per l'esecuzione delle analisi appena presentate; Tale scelta non è casuale in quanto *Akoma Ntoso*, grazie a importanti caratteristiche quali la sua flessibilità e la sua universalità, ha avuto un notevole successo negli anni ed è tuttora utilizzato in diversi contesti mondiali. Un'altra peculiarità di questo standard che ha influito nella scelta, è la sua ricchezza semantica, caratteristica molto utile poiché permette di eseguire navigazioni e analisi complesse e specifiche sul documento tramite l'utilizzo di strumenti appropriati. Per la trattazione completa di *Akoma Ntoso* si rimanda alla sezione 4.3.

5.2 Dashboard e Data Visualization

Come strumento per la realizzazione delle visualizzazioni dei risultati ottenuti dalle analisi ho deciso di sviluppare una dashboard poiché tale strumento permette di comunicare in modo semplice e intuitivo le informazioni che si vogliano mostrare. La dashboard, in accordo con Stephen Few [Few06], è una visualizzazione grafica delle informazioni più importanti, le quali sono necessarie per raggiungere uno o più obiettivi; tali informazioni sono organizzate e strutturate in modo da renderle accessibili e fruibili da un punto unico. La dashboard può fornire un modo potente per presentare delle informazioni però spesso falliscono nell'obiettivo di comunicare qualcosa in modo efficace ed efficiente a causa di un design inadeguato e povero. Quindi si può parlare del successo di una dashboard, come mezzo di comunicazione, se quest'ultima riesce a presentare le informazioni in modo chiaro e immediato.

5.2.1 Funzionalità di SOFIA

La dashboard deve mostrare in modo piacevole e intuitivo i risultati delle analisi, e mettere a disposizione dell'utente finale che può essere un data journalist o un semplice lettore, un insieme di funzionalità che hanno l'obiet-

tivo di fornire uno strumento completo e flessibile. Le funzionalità presenti in *SOFIA* sono:

- possibilità di scegliere il database da utilizzare per il recupero dei documento; le tipologie di db che si vogliono supportare sono: file system; eXist-db che sarà presentato nella sezione 6.4.1;
- visualizzare la lista di tutti i documenti *Akoma Ntoso* presenti nel database selezionato;
- visualizzare il contenuto dei documenti;
- scaricare in locale uno o più documenti selezioni.
- aggiungere documenti *Akoma Ntoso* al database tramite il caricamento di singoli file o di file zip che contengono uno o più documenti;
- possibilità di selezionare la rappresentazione grafica da visualizzare;

Figura 5.1: Interfaccia di *SOFIA*

La figura 5.1 mostra l'interfaccia finale della dashboard sviluppata. Nella parte alta della dashboard sono presenti i due campi per la gestione e la

selezione del database da utilizzare. Sulla sinistra si può notare il numero totale dei documenti caricati insieme alla lista di quest'ultimi, oltre che i due pulsanti per il caricamento di nuovi documenti e il download di quelli selezionati. La parte centrale della dashbord rappresenta il piano di lavora dove possono essere presentati i grafici (caso mostrato in figura) oppure dove poter visualizzare il contenuto di ogni documento presente nella lista sulla sinistra. La selezione del grafico da visualizzare può essere fatta attraverso il pulsante "Show Chart" che presenta una lista con le varie opzioni. Il pulsante "Clear" permette di chiudere quello che è attualmente visualizzato nella parte centrale.

5.2.2 Componenti necessarie

Sono state discusse le funzionalità che si vogliono implementare in *SOFIA*, adesso saranno analizzate le componenti necessarie per il suo sviluppo. La principale componente è Node.js, descritto nella sezione 6.4.1, ambiente su cui sono stati sviluppati il lato client e il lato server di questa web applicazion, utilizzando solamente JavaScript[3]. Entrambi questi aspetti, lato client e lato server, della dashboard saranno trattati di seguito mettendo in evidenza le macro aree di interesse e le principali componenti che permettono la loro realizzazione.

Client. Il lato client è responsabile dell'interfaccia utente che è stata sviluppata con il framework semantic-ui[4], il quale permette di creare layout accattivanti e responsive. Un aspetto molto importante per uno strumento che mira a visualizzare i risultati di un'analisi è la libreria per le rappresentazioni grafiche; per questo scopo è stata utilizzata Chart.js[5], libreria JavaScript che permette di creare diverse tipologie di grafici in modo semplice e flessibile. Infine, prevedendo l'implementazione delle analisi nel client, è necessario avere a disposizione una libreria utilizzabile in ambiente browser che permetta di navigare ed estrarre informazioni dai documenti *Akoma Ntoso*. Nella sezione 5.3 vedremo *akomando*, una libreria per estrapolare delle informazioni

[3] `https://www.javascript.com/`
[4] `https://semantic-ui.com/`
[5] `https://www.chartjs.org/`

dai singoli documenti *Akoma Ntoso* che va in direzione delle esigenze della dashboard.

Il client comunicherà con il server, tramite chiamate AJAX[6], per richiedere documenti e per il loro caricamente nel database.

Server. Il lato server si occuperà dell'interfacciamento con le due tipologie di database previste dalle funzionalità di *SOFIA*: fs e eXist-db. Questo aspetto è fondamentali sia per l'esecuzione delle analisi in quanto *akomando* lavora solamente in locale, sia per la gestione di alcune funzionalità come la visualizzazione del contenuto dei singoli documenti. Le interfacce con i database dovranno fornire il supporto necessario per il recupero dei documenti e per la loro aggiunta nel database. La fase di recupero deve prevedere anche la possibilità di scaricare più documenti contemporaneamente. La fase di archiviazione invece deve fornire all'utente la possibilità di caricare file zip contenenti documenti *Akoma Ntoso*. L'interazione con i database è un aspetto che sarà ripreso nella sezione 5.4.

La figura 5.2 mostra un'astrazione delle interazioni tra i principali moduli che compongono *SOFIA*. Tali moduli sono presentati con diversi colori: verde per indicare quelli già discussi che non hanno bisogno di ulteriori precisazioni; arancione per indicare *akomando*, modulo che verrà presentato in maniera completa nella sezione 5.3; rosso per indicare la mancanza di strumenti per la realizzazione del modulo, tale aspetto sarà ripreso nella sezione 5.4 e verrà proposta una soluzione nel capitolo 6.

5.3 Akomando: libreria document-centered per Akoma Ntoso

Akomando[7] è una libreria JavaScript open source per la gestione dei documenti *Akoma Ntoso*. Tale libreria permette di lavorare con un singolo documento in locale e con accesso in sola lettura. *Akomando* è caratterizzato da un insieme di funzionalità che consentono la navigazione e l'analisi del

[6] `https://www.w3schools.com/xml/ajax_intro.asp`
[7] `https://www.akomando.bitnomos.eu/`

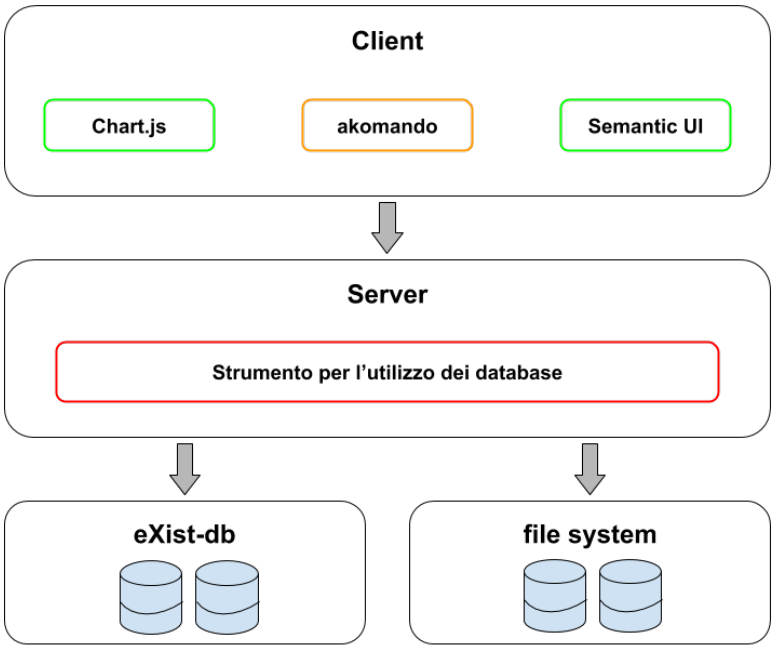

Figura 5.2: Astrazione dei moduli principali che compongono *SOFIA*

contenuto di un documento *Akoma Ntoso* e anche funzionalità specifiche per l'estrazione di informazioni. *Akomando* è stata progettata e sviluppata da BitNomos[8].

Figura 5.3: Akomando

[8] `https://www.bitnomos.eu/`

Modalità di utilizzo. *Akomando* permette tre modalità d'accesso per la fruizione delle sue funzionalità, questo lo rende una libreria molto dinamica e utilizzabile in diversi contesti. Le tre modalità di utilizzo sono:

- ambiente Node.js, che introdurremo nella sezione 6.4.1;
- ambiente browser, importando tale libreria nel file HTML;
- interfaccia a linea di comando specificando il comando da eseguire e il file su cui si vuole lavorare.

5.3.1 Funzionalità e API

Le API per le funzionalità di *akomando* sono accessibili non appena l'utente richiama la funzione "createAkomando" passando il file *Akoma Ntoso* codificato in stringa e, potenzialmente, un file di configurazione per la gestione delle diverse versione dello standard *Akoma Ntoso*. Di seguito sono mostrate le modalità per includere la libreria *akomando* e l'utilizzo di "createAkomando" per la sua istanziazione.

```
import { createAkomando } from 'akomando'; //ES6
var createAkomando = require('akomando').createAkomando; //ES5

const akomandoApi = createAkomando({
   aknString: AkomaNtosoString, //stringa contenente il file akn
});
```

A questo punto abbiamo nella variabile "akomandoApi" l'istanza della libreria *akomando* che ci permette di accedere alle sue funzionalità che saranno descritte brevemente di seguito.

getAkomaNtoso. Funzione che permette di richiedere l'intero documento *Akoma Ntoso* serializzato in un certo formati, ad esempio oggetto DOM, stringa XML, JSON, testo semplice. Inoltre con una serializzazione HTML possono essere utilizzati altri parametri che permettono di ricevere, ad esempio, solo il body o solo l'elemento head del documento.

getDocType. getDocType restituisce il tipo del documento *Akoma Ntoso*, ad esempio act, bill, documentCollection.

getMeta. getMeta restituisce tutti i metadati presenti nel documento; è possibile specificare come devono essere serializzati i risultati che si vuole ricevere.

getHierIdentifier. Tale funzione ritorna un oggetto contenente tutti gli identificatori degli elementi gerarchici, ad esempio: article, paragraph. Tramite i parametri accettati da questa funzione è possibile filtrare le gerarchie ritornate specificando il nome che devono avere oppure il loro contenuto.

countDocElements. Tale funzione ritorna un oggetto con le informazioni riguardanti il numero di ogni elemento contenuto nel documento *Akoma Ntoso*. Tramite i parametri è possibile ricevere informazioni riguardanti gli elementi con un certo nome.

getReferencesInfo. getReferencesInfo ritorna un oggetto con la lista dei riferimenti presenti nella sezione metadati (tag "meta") del documento. Inoltre vengono ritornate informazioni riguardanti i singoli riferimenti e gli elementi che sono collegati ad essi.

getComponents. getComponents restituisce una lista contenente le informazioni che riguardano le componenti presenti nel documento. Tramite i parametri è possibile decidere la serializzazione dei dati ritornati.

5.3.2 Utilizzo nella dashboard

Akomando ha un ruolo fondamentale nella dashboard *SOFIA* in quanto ci permette di estrapolare informazioni dai documenti *Akoma Ntoso* che saranno alla base delle analisi e quindi delle visualizzazioni.

Infatti per sviluppare alcune analisi definite in precedenza è stata utilizzata la funzione "getHierIdentifier" che permette di estrarre dal documento tutti gli elementi di interesse ovvero gli articoli e i paragrafi. Per altre esigenze relative allo sviluppo delle analisi, invece, è stato necessario implementare delle nuove funzionalità nella libreria *akomando*.

Nuove funzionalità per akomando. Le informazioni riguardanti gli articoli e i paragrafi devono essere raggruppate per anni e per legislatura. Tale aspetto ha fatto emergere la mancanza di una funzione che restituisca le informazioni relative all'autore e all'anno di pubblicazione del documento legislativo, basate sempre sul modello FRBR. A tale fine ho proposto e implementato in *akomando* la funzione "getPublicationInfo" che accetta come parametro una stringa e che permette all'utente di definire l'entità (work, expression, manifestation) di cui necessita delle informazioni. Segue un esempio dell'oggetto ritornato da questa funzione.

```
{
        "author" : {
                "name":
                "ref":
        },
        "date":
}
```

In secondo luogo, è stato necessario proporre e implementare una funzionalità aggiuntiva all'interno della funzione "getAkomaNtoso". Questa necessità è dovuta alla mancanza, in *akomando*, di un meccanismo per recuperare, dato un documento *Akoma Ntoso*, il testo relativo al solo elemento body.

5.4 Dal documento singolo alla collezione di documenti

Come accennato nella sezione 5.2.2, ci si è resi conto che non esiste nessuno strumento che permetta in modo semplice e aderente allo standard *Akoma Ntoso*, di archiviare e recuperare documenti da un database. Quindi una prima soluzione potrebbe essere l'implementazione delle due interfacce per eXist-db e file system. Questa soluzione risulta però costosa poiché è necessario studiare la libreria fs[9] per la gestione del file system e, per eXist-db bisogna studiare i modi e i mezzi per richiedere o caricare risorse. Un'ulteriore difficoltà è data dalla struttura che si vuole dare ai database al fine di ren-

[9] `https://nodejs.org/api/fs.html`

derli semanticamente comprensibili e soprattutto per archiviare i documenti in modo univoco utilizzando un concetto alla base della naming convention di *Akoma Ntoso*, ovvero il modello FRBR.

Nel prossimo capitolo sarà presentato il contributo scientifico del mio progetto di tesi che permette il completamento di *SOFIA* tramite uno strumento che ne facilità la costruzione. Nel capitolo 7 saranno presentati i risultati ottenuti dalle analisi presentate, tramite delle rappresentazioni grafiche.

Capitolo 6

Akomando-db: tool per la gestione dei documenti Akoma Ntoso

Nel capitolo 5 è stata concettualizzata *SOFIA*, una dashboard che ha l'obiettivo di presentare graficamente le analisi su documento giuridici in formato *Akoma Ntoso*. Durante questa descrizione è emersa la necessità di avere uno strumento che permettesse in modo semplice e coerente con lo standard *Akoma Ntoso*, azioni per l'archiviazione e il recupero di documento da un database, indipendentemente dal tipo di database utilizzato. Durante questo capitolo verrà descritto il risultato finale del mio progetto di tesi, *akomando-db*. Tale strumento nasce da un'esigenza ma è sviluppato in modo indipendenda da essa poiché va a colmare un vuoto dato dalla mancanza di strumenti per lo standard *Akoma Ntoso* per la gestione del ciclo di vita di un documento, visto come archiviazione, utilizzo e recupero.

In questo capitolo sarà descritta la libreria *akomando-db* che permette di semplificare e standardizzare a chi lavorare con i documento *Akoma Ntoso*, la gestione delle attività connesse con l'archiviazione e il recupero dei documenti. Con questa libreria l'utente ha a disposizione uno strumento ricco di funzionalità e di controlli per la gestione degli errori, utilizzabile tramite API e tramite interfaccia a riga di comando. In questo modo è possibile impiegare meno tempo e meno risorse per la gestione dei diversi database, e concentrar-

si maggiormente sulla logica delle applicazioni o dei servizi che si vogliono sviluppare in un qualsiasi contesto sia richiesto l'utilizzo di documenti *Akoma Ntoso*.

6.1 Obiettivi

Nella sezione 4.3.3 è stata introdotta la naming convention di *Akoma Ntoso* e il modello FRBR su cui si basa l'identificazione delle risorse. Tale modello garantisce significatività e univocità agli identificatori delle risorse, caratteristica utile per la gestione dei database, ma aggiunge un livello di difficoltà per chi deve utilizzarlo concretamente. Questa difficoltà è data dalla comprensione delle tre entità e dal loro utilizzo a secondo dell'obiettivo che si intende raggiungere.

Akomando-db è stato pensato e sviluppato con tre obiettivi principali:

- sfruttare il modello concettuale FRBR su cui si basa la naming convention di *Akoma Ntoso* per la gestione delle attività legate all'archiviazione e al recupero dei documenti; questo ha l'obiettivo di garantire una strutturazione significativa dei database e, contemporaneamente, rimanere aderenti allo standard. Inoltre *akomando-db* fornisce un punto d'accesso normalizzato ai database per la gestione dei documenti *Akoma Ntoso* che ha l'obiettivo di sfruttare il suo orientamento universale e permettere riferimenti corretti e permanenti tra diversi documenti in diversi contesti;

- semplificare le attività dell'utente per quanto riguarda la gestione di diversi database così da rendere superfluo lo studio dei meccanismi e delle API specifici per ognuno di essi. *Akomando-db* offre delle API che permettono di gestire archiviazione e recupero dei documenti indipendentemente dal database sottostante che può essere di una tipologia piuttosto che di un'altra;

- aggiungere uno strumento che semplifichi certe attività, agli strumenti già esistenti per *Akoma Ntoso* al fine di incentivarne la diffusione e l'utilizzo.

Oltre alle funzionalità che sono emerse da questi tre obiettivi principali, *akomando-db* offre un insieme di funzionalità aggiuntive, descritte di seguito, che facilitano la vita dell'utilizzatore per quanto riguarda alcune attività collegate alle operazioni di archiviazione e di recupero, come ad esempio la richiesto di file zip oppure la gestione degli allegati collegati ai documenti *Akoma Ntoso*.

6.2 Casi d'uso

Per comprendere meglio l'utilità e il contributo fornito dalla libreria *akomando-db*, di seguito sarà descritto un caso d'uso principale e alcuni secondari. Il caso d'uso principale è necessario per la verifica della tesi presentata nell'introduzione e sarà ripreso nella sezione 7.2. I casi d'uso secondari non saranno approfonditi nella dissertazione ma hanno un ruolo importante per chiarire al lettore le varie attività dov'è possibile utilizzare *akomando-db*.

- **caso d'uso principale**: Sabrina è una donna di 40 anni con un background formativo in informatica e che da 8 anni lavora nell'ambito del Data Journalism. Nel contesto di un progetto per un ente pubblico italiano, utilizzando un insieme di documenti del sistema legislativo italiano nello standard *Akoma Ntoso*, deve fornire un resoconto riguardo l'organizzazione strutturale dei documenti nei diversi periodi e nelle diverse legislature. L'idea è quella di creare una dashboard, in Node.js, per sviluppare delle visualizzazioni che mostrino i risultati ottenuti dalle analisi fatte. In questa dashboard dovrà essere possibile scegliere il database da utilizzare e caricare nuovi documenti nel database; queste due funzionalità sono necessarie per permette all'ente pubblico di sfruttare la dashboard anche in altri contesti e con documenti diversi da quelli di partenza. Sabrina conosce la libreria *akomando* che è utile per la gestione e l'estrapolazione di informazioni dai documenti *Akoma Ntoso* presenti in locale ma non ha tempo per studiare i vari strumenti e i vari meccanismi necessari alla gestione dei documenti sul database eXist, indicato dall'ente pubblico come database di base per il progetto. Quindi inizia a ricercare uno strumento che le consenta di implemen-

tare tale attività in modo semplice e intuitivo. Potrebbe utilizzare la libreria *akomando-db* che permette di eseguire l'interfacciamento con il database eXist-db utilizzando delle semplici funzioni di API;

- **caso d'uso secondario**: Flavio, 55 anni, è un professore di Informatica Giuridica dell'università di Udine. Ha assegnato un progetto ai suoi studenti riguardante lo studio e l'analisi dello standard *Akoma Ntoso*. Flavio decide di dare un file zip con un centinaio di documenti ai ragazzi prendendoli da un database eXist-db dell'ateneo. Per svolgere questa attività può utilizzare la libreria *akomando-db* da linea di comando per richiedere facilmente il recupero di tutti i file presenti nel database in un unico file zip;

- **caso d'uso secondario**: Davide è uno studente di informatica, 25 anni, che ha vinto un bando presso la scuola di scienze. Per questo bando deve mettere mano ad un software che si chiama *akomando-db* al fine di aggiungere il supporto a IPFS (InterPlanetary File System), protocollo per l'archiviazione distribuita di dati. Dopo una prima analisi, Davide scopre che *akomando-db* è modulare e che quindi non dovrà mettere mano a file scritti due-tre anni prima da altri, ma basterà aggiungere un modulo che si interfacci alle API di IPFS e sperare che tutte le funzioni di *akomando-db* lavorino come descritto nella documentazione ufficiale;

- **caso d'uso secondario**: Sofia è una ragazza di 30 anni dipendente presso un ente privato che si occupa di supporto e sviluppo delle attività connesse all'Informatica Giuridica. Questo ente lavora con documenti *Akoma Ntoso* e quindi necessità di strumenti per archiviare e recuperare documenti in tale formato che possono trovarsi sul file system o su database eXist-db. Sofia, per testare i vari strumenti che sviluppa e mantiene, ha bisogno frequentemente di nuovi documenti *Akoma Ntoso* nei vari database; attualmente ha creato un piccolo script che carica i documenti nei vari database senza controlli e senza strutturarli. Potrebbe utilizzare *akomando-db*, sviluppato per *Akoma Ntoso*, al fine di velocizzare, standardizzare e controllare tale attività.

6.3 Funzionalità e utilizzo tramite API

Akomando-db è una libreria asincrona scritta in JavaScript e utilizzabile in ambiente Node.js, strumento descritto in seguito. Tale libreria nasce come estensione della libreria *akomando* (sezione 5.3) per la gestione delle collezioni di documenti *Akoma Ntoso*; infatti *akomando* non lavora su collezioni di documenti bensì solamente su documenti singoli presenti in locale.

Akomando-db offre una semplice API che si mappa perfettamente con le sue principali funzionalità e che presenta un insieme di parametri settabili dall'utente, i quali hanno l'obiettivo di individuare le specifiche attività richieste. Per utilizzare la libreria è necessario importarla tramite uno dei modi seguenti:

```
var akomandoDB = require('akomando-db'); // ES5
import { akomandoDB } from ( 'akomando-db' ) ; // ES6
```

Akomando-db si presenta come una classe che deve essere istanziata tramite un apposito costruttore e, una volta creata l'istanza, è possibile utilizzare le funzionalità base messe a disposizione dell'utente. Segue la riga di codice necessaria per istanziare la libreria.

```
var aknDB = new akomandoDB();
```

In questa sezione vedremo, ad alto livello, i tre aspetti base di *akomando-db*, il mapping con le API, e i diversi settaggi possibili tramite i parametri. La descrizione architetturale e implementativa di *akomando-db* sarà trattata nella sezione 6.4.

6.3.1 Inizializzazione del database

Il primo passo per utilizzare *akomando-db* è inizializzarlo con le informazioni relative al database che si vuole utilizzare. Questa inizializzazione può essere fatta direttamente con il costruttore o utilizzando un metodo apposito chiamato "setDB", come mostrato di seguito:

```
aknDB.setDB({
        db: "exist",
        host: "127.0.0.1",
        port: 8080,
```

```
        entry_point: "/exist/rest",
        user: "admin",
        psw: "admin"
});
```

Attualmente *akomando-db* gestisce due tipi di database, file system ed eXist-db. Nel frammento di codice mostrato sopra è stato inizializzato un database eXist-db per *akomando-db*, invece per inizializzare un database file system sono necessari i seguenti parametri:

```
db: "fs",
path: "./test/db_akomando"
```

A questo punto l'istanza di *akomando-db* che risiede nella variabile "aknDB" è già pronta per l'utilizzo oppure è in uno stato di errore se qualcosa nell'inizializzazione del database non è andata a buon fine. Per visionare lo stato dell'istanza di *akomando-db* e altre informazioni relative a quest'ultima, sono messe a disposizione dell'utente una serie di campi; segue la descrizione di quest'ultimi insieme ai valori possibili.

```
aknDB.state = active | error
aknDB.error =<descrizione dell'errore>
aknDB.db_type = fs | exist
```

6.3.2 Archiviazione nel database

Le funzionalità per l'archiviazione e per il recupero dei documenti sono fortemente influenzate dal modello concettuale FRBR che è alla base della naming convention per l'identificazione delle risorse. Infatti, utilizzando *akomando-db*, la struttura interna del database che si intende utilizzare seguirà la struttura degli FRBR utilizzata dai documento *Akoma Ntoso*. Quella che segue è la descrizione dell'API per l'aspetto collegato all'archiviazione dei documenti:

```
### i valori gia' inseriti indicano i valori di default
aknDB.push({ par, push_type='manifestation', zip=false,
    attachment_name, attachment_path='', force=false }, callback)

### lista dei parametri con i tipi associati
par = {(String|String[]|buffer)}
```

```
push_type = {String}
zip       = {Boolean}
attachment_name = {String|String[]}
attachment_path = {String}
force = {Boolean}
```

Il parametro "push_type" ci permette di definire che tipo di documento si vuole salvare nel database e garantisce l'aderenza al modello FRBR. I possibili valori per tale parametro sono:

- **work**: permette di inserire un work nel database che può essere visto come un contenitore o un folder contenente delle expression;
- **expression**: permette di inserire un'expression nel database che può essere vista come un contenitore o un folder contenente delle manifestation; la creazione di un'expression prevede la creazione implicità del work relativo, se non esiste;
- **manifestation**: permette di inserire la manifestation di un documento, ovvero il file XML *Akoma Ntoso* rappresentante il documento; questa tipologia di archiviazione prevede la creazione dei relativi work ed expression al fine di mantenere una struttura nel database che rispecchi quella dell'iri FRBR relativo alla manifestation;
- **attachment**: permette il caricamento di un allegato ovvero un documento non necessariamente *Akoma Ntoso*.

Il parametro "par" serve per poter passare i file da archiviare: nel caso di "attachment" saranno richiesti dei file codificati in binario; nel caso di "manifestation" saranno richiesti i file codificati come stringhe; se invece si vuole aggiungere un "work" oppure una "expression" sarà necessario specificare gli FRBR da aggiungere. Segue un breve esempio per chiarificare il mapping tra il modello FRBR (work, expression, manifestation) e la struttura interna del database. Ipotizzando un file system come db, definiamo una cartella(./db) inizialmente vuota, come entry point, ed eseguiamo tre azioni in sequenza. Nell'esempio, tra un'azione e l'altra viene mostrato lo stato del database.

```
### STATO DB identifica lo stato interno dell'ipotetica database
```

```
### prima azione
aknDB.push({par:"/akn/eu/jugment/summary/unibo/2014-06-12/C
    -39-13", push_type:"work"}, ()=>{})
STATO DB: ./db/akn/eu/jugment/summary/unibo/2014-06-12/C-39-13

### seconda azione
aknDB.push({par:"/akn/eu/jugment/summary/unibo/2014-06-12/C
    -39-13/ita@", push_type:"expression"}, ()=>{})
STATO DB: ./db/akn/eu/jugment/summary/unibo/2014-06-12/C-39-13/
    ita@

### terza azione
aknDB.push({par: <file in formato stringa>, push_type:"
    manifestation"}, ()=>{})
STATO DB: ./db/akn/eu/jugment/summary/unibo/2014-06-12/C-39-13/
    ita@/main.xml
```

I tre passaggi appena mostrati sono implicitamente richiamati non appena si esegue la push manifestation (ovvero il comando push con il parametro push_type con valore manifestation), questo permette di evitare un lavoro inutile e ripetitivo all'utente. Si noti come l'entry point del file system, nell'esempio, segua perfettamente la struttura dell'FRBR. Questa strutturazione rende più chiara l'organizzazione interna dei database e consente di mantenere un alto livello di aderenza con il modello FRBR.

Settando il parametro "zip" con il valore true è possibile passare nel parametro "par" un file zip codificato in binario, contente uno o più documenti *Akoma Ntoso*. Tale funzione caricherà nel database tutti i file *Akoma Ntoso* ben formati, presenti nel file zip, scartando tutti i file di altri formati. I parametri "attachment_name" e "attachment_path" sono strettamente collegati alla push attachment (ovvero con parametro push_type uguale ad attachment); essi infatti permettono di definire il nome con cui il file deve essere salvato nel database e la posizione(expression) dove archiviarlo. Il parametro "force" permette di sovrascrivere i file nel caso in cui questi siano già presenti nel db.

L'utente nel momento dell'utilizzo dell'API dovrà definire anche una callback che verrà richiamata passando una lista di oggetti dove ognuno di essi contiene due campi:

- la lista “infoList” contiene informazioni più o meno rilevanti sui passaggi che sono stati effettuati dalla libreria;

- “error” contiene l’errore che non ha permesso di portare a buon fine l’esecuzione, oppure nel caso di un file zip, la lista di errori che non hanno permesso il corretto funzionamento relativo a determinati documenti.

Si è deciso di ritornare una lista di oggetti e non un singolo oggetto poiché l’utente può definire più azioni con una singola chiamata e quindi la lunghezza della lista è uguale al numero di valori passati al parametro “par” che corrisponderà al numero di azioni richieste. Di seguito un esempio della lista di oggetti ritornata.

```
[
        {
                error: <lista di errori o null>
                infoList: <lista di informazioni (anche vuota)>
        },
        { ... },
        { ... }
]
```

6.3.3 Recupero dal database

Come per le funzionalità di archiviazione, quelle per il recupero dei documenti sono fortemente influenzate dal modello concettuale FRBR che è alla base della naming convention per l’identificazione delle risorse. Quella che segue è la funzione dell’API relativo all’aspetto collegato al recupero dei documenti:

```
### i valori gia' inseriti indicano i valori di default
aknDB.pull({ par=null, pull_type='manifestation', zip=false, all
    =false }, callback)

### lista dei parametri con i tipi associati
par = {(String|String[])}
pull_type = {String}
zip = {Boolean}
all = {Boolean}
```

Il parametro "par" identifica la lista di FRBR che rappresentano i file che si intende scaricare. Il parametro "pull_type" permette invece di definire il tipo di operazione "pull" da eseguire:

- **work**: ritorna la lista delle expression presenti nell'FRBR work passato in "par";
- **expression**: ritorna la lista dei file (*Akoma Ntoso* e allegati) presenti nell'FRBR expression passata in "par";
- **manifestation**: ritorna il buffer codificato in binario che rappresentano il file individuato con l'FRBR manifestation passata in "par";
- **attachment**: ritorna la lista di allegati presenti nell'FRBR expression passata in "par".

Se l'utente cerca di utilizzare un certo tipo di "pull" (work, expression, manifestation) passando un FRBR che non rispecchia il valore della pull_type, *akomando-db* ritornerà un errore all'utente. L'esempio seguente cerca di chiarire la differenza tra i diversi valori possibili del parametro "pull_type". Definito il file system come database e un entry point (./db) già popolato si eseguono le seguenti tre azioni.

```
### STATO INIZIALE DEL DB (./db):
./db/akn/eu/jugment/summary/unibo/2014-06-12/C-39-13/ita@/!main.
    xml
./db/akn/eu/jugment/summary/unibo/2014-06-12/C-39-13/eng@/!main2
    .xml

### prima azione
aknDB.pull({par:"/akn/eu/jugment/summary/unibo/2014-06-12/C
    -39-13", pull_type:"work"}, ()=>{})
### RISULTATO: [ita@, eng@]

### seconda azione
aknDB.pull({par:"/akn/eu/jugment/summary/unibo/2014-06-12/C
    -39-13/ita@", pull_type:"expression"}, ()=>{})
### RISULTATO: [main.xml]

### terza azione
```

```
aknDB.pull({par: "/akn/eu/jugment/summary/unibo/2014-06-12/C
    -39-13/eng@/!main2.xml", pull_type:"manifestation"}, ()=>{})
### RISULTATO: <file binario che rappresenta il documento "main2
    .xml">
```

Gli esempi appena trattati mostrano i comportamenti di default della funzione "pull" che possono essere modificati andando a settare i due parametri seguenti:

- zip: settando a true tale parametro è possibile far si che tutti i comandi che restituiscono una lista di FRBR di default, restituiscano invece un file zip contenente i documenti rappresentati nel database dagli FRBR presenti nella lista. Il file zip viene ritornato al chiamante come buffer codificato in binario;

- all: tale parametro permette di ritornare la lista di tutte le manifestation o di tutti gli allegati presenti nel DB.

La callback definita dall'utente verrà chiamata passando una lista di oggetti dove ogni oggetto contiene tre campi: result, error, infolist. Viene ritornata una lista di oggetti poiché l'utente può definire più azioni con una singola chiamata e quindi la lunghezza della lista è uguale al numero di valori passati al parametro "par" che corrisponderà al numero di azioni richieste. Di seguito un esempio della lista di oggetti ritornata.

```
[
        {
                result: <lista di FRBR o lista di buffer
                    codificati in binario o null>
                error: <lista di errori o null>
                infoList: <lista di informazioni (anche vuota)>
        },
        { ... },
        { ... }
]
```

6.3.4 Utilizzo tramite interfaccia a linea di comando

Tutte le funzionalità viste nella sezione precedente sono utilizzabili da API ma anche da interfaccia a linea di comando (CLI). L'idea base è quella di avere due comandi "push" e "pull" e un insieme di flag che permettono di mappare tutte i parametri delle API viste sopra. I flag previsti sono i seguenti: output, db, work, expression, manifestation, attachment, attachment_name, attachment_path, zip, all, force. Questi flag si mappano perfettamente con i parametri discussi precedentemente, le uniche eccezioni che bisogna attenzionare sono:

- per l'inizializzazione del database bisogna utilizzare il flag "–db" passando un file json che definisce i parametri per il db; quindi ogni comando, utilizzato da linea di comando, inizializzerà *akomando-db* per poi eseguire l'operazione richiesta;

- con il flag "–output" possiamo definire dove dovrà essere salvare il file scaricare dal database.

L'esempio seguente mostra alcuni mapping tra i flag della CLI e i parametri dell'API ipotizzando di avere un file "db_config.json" dove sono definiti i parametri per l'inizializzazione del database in *akomando-db*:

```
aknDB.push({par:"/akn/it/doc/2017-09-18/", push_type:'work', zip
    :false, force:false }..)
akomando-db push --work /akn/it/doc/2017-09-18/ --db ./config_fs
    .json

aknDB.pull({par:"/akn/it/doc/2017-09-18/ita@/", pull_type:'
    attachment', zip:false, all:false }..)
akomando-db pull --attachment /akn/it/doc/2017-09-18/ita@/ --db
    ./config_fs.json

aknDB.pull({par:null, pull_type:'manifestation', zip:false, all:
    true }..)
akomando-db pull --all --db ./config_fs.json
```

6.4 Sviluppo di Akomando-db

In questa sezione saranno descritte le principali librerie che sono state utilizzate per lo sviluppo di *akomando-db*, la sua architettura e le componenti che lo costituiscono.

6.4.1 Librerie e tecnologie utilizzate

Prima di descrivere l'architettura di *akomando-db* definiamo alcune delle componenti principali che sono state utilizzate per tutta la fase di sviluppo di *akomando-db*.

eXist-db. eXist-db[1] o eXist è un progetto open source costruito su tecnologie XML, creato nel 2000. eCist-db è classificato sia come database NoSQL document-oriented, sia come database XML nativo. Esso fornisce gli strumenti XQuery e XSLT per la definizione di query e inoltre supporta l'interazione tramite chiamate REST[2].

Node.js. Node.js[3] è un runtime JavaScript costruito sul motore JavaScript V8 di Google Chrome. Node.js rende possibile l'esecuzione di applicazioni JavaScript in locale e nei server, quindi fuori dall'ambiente dei browser. Questo permette di creare applicazioni full-stack con un unico linguaggio.

npm. npm[4] è il principale gestore di pacchetti per JavaScript. E' possibile utilizzare npm tramite l'interfaccia a linea di comanda che è inclusa in Node.js. Tale gestore è stato utilizzato per installare tutte le librerie utilizzate durante lo sviluppo di *akomando-db*.

Babel. Babel[5] è un compilatore JavaScript utilizzato principalmente per convertire il codice JavaScript conforme a ECMAScript 2015 o successivo in

[1] `http://exist-db.org/exist/apps/homepage/index.html`
[2] `https://en.wikipedia.org/wiki/EXist`
[3] `https://nodejs.org/en/`
[4] `https://www.npmjs.com/`
[5] `https://babeljs.io/`

una versione precedente e compatibile con i browser e gli ambienti come Node.js che non accettato ancora le nuove feature incluse negli ultimi standard. Babel è fondamentale in *akomando-db* che utilizza le sintassi e le feature più recenti di JavaScript, come ad esempio la "import" per includere le librerie, che non sono ancora supportate da Node.js.

6.4.2 Unit testing

Un aspetto molto interessante che ha accompagnato tutto lo sviluppo di *akomando-db* è stato l'utilizzo degli unit test. Questi rappresentano un metodo di testing col quale le singole unità del codice sorgente sono testate per validarne la correttezza. Tale strumento permette una maggiore mantenibilità del codice, oltre che aiutare lo sviluppatore nell'individuazione degli errori. Per la loro realizzazione sono state utilizzare due librerie:

- Mocha[6]: framework per la definizione di test in JavaScript che lavora su Node.js e sui browser. Mocha però non ha una libreria per le asserzioni e quindi può essere integrata con una libreria come Chai.

- Chai[7]: Chai è una libreria per la definizione di asserzioni per Node.js e per i browser che può essere abinata ad un qualsiasi framework per il testing di codice JavaScript.

Tramite l'utilizzo combinato di questi due strumenti sono stati definiti un insieme di unit test per i comandi dell'interfaccia a linea di comando (59 unit test) e per le API (52 unit test). Tali test si dividono in due parti, metà per verificare il funzionamente di eXist-db, l'atra metà per il funzionamento del file system. La creazione degli unit test è stata costante in modo tale da avere più di un test per ogni funzionalità implementata. Seguono alcuni esempi di unit test per l'interfaccia a linea di comando e per le API.

```
### esempio di unit test per CLI (command line interface)
it('should return a list containing 3 item: 2 attachments and 1
    xml; passing one expression', (done) => {
```

[6] https://mochajs.org/
[7] https://www.chaijs.com/

```
        exec("babel-node ./dist/akomando-db.bin.bundle.js pull
            --expression /akn/it/doc/2017-09-18/ita@ --db ./
            config_fs.json", (error, stdout, stderr) => {
                expect(stdout).contain('banner.jpg')
                expect(stdout).contain('config.json')
                expect(stdout).contain('main.xml')
                expect(stderr).equal('');
                done();
        })
}).timeout(4000);

### esempio di unit test per le API
it('should return a list containing 3 item, 2 attachments and 1
    xml; passing one expression', (done) => {
        var akomandoDb = new AkomandoDB()
        akomandoDb.setDB(config_file_fs)
        akomandoDb.pull({par:["/akn/it/doc/2017-09-18/ita@"],
            pull_type:"expression"}, (result) => {
                result = result[0]
                expect(result.result).to.be.an('array').that.
                    includes('config.json');
                expect(result.result).to.be.an('array').that.
                    includes('banner.jpg');
                expect(result.result).to.be.an('array').that.
                    includes('main.xml');
             expect(result.error).to.be.a('null')
             done();
    });
}).timeout(4000);
```

6.4.3 Architettura e implementazione

Akomando-db presenta un'architettura modulare, semplice ed estendibile. Un'astrazione di quest'ultima è mostrata nella figura 6.1, dal livello più alto caratterizzato dalle API fino al livello più basso caratterizzato dall'interfaccia verso i database. *Akomando-db* è una libreria asincrona e quindi tutte le operazioni prevedono il passaggio di una callback da chiamare una volta che l'esecuzione è terminata; questo approccio permette un utilizzo dinamico e non bloccante della libreria.

In questa sezione saranno trattati i moduli principali, descrivendo il loro ruolo, le relazioni con i moduli collegati e le informazioni implementative più interessanti o complesse. Inoltre verrà presentata una funzione integrata nella libreria *akomando* in quanto necessaria al funzionamente di *akomando-db* ma concettualmente appartenente alle funzionalità di *akomando*. I paragrafi

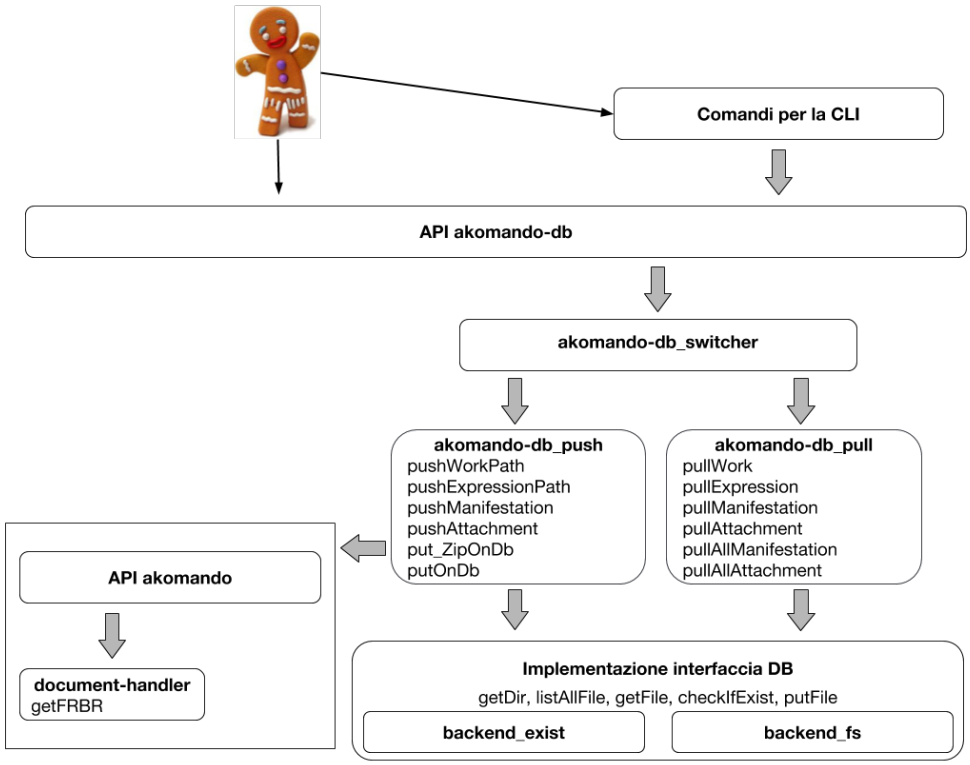

Figura 6.1: Architettura *akomando-db*

seguenti riprendono i singoli moduli dell'astrazione presentata nella figura 6.1 così da mantenere un mapping forte tra moduli trattati e le relazioni tra i vari moduli che compongono *akomando-db*.

Comandi per la CLI. Tramite questo modulo è possibile utilizzare le funzionalità di *akomando-db* con la CLI (command line interface), appoggiandosi alle API di *akomando-db*. La libreria che ha permesso la sua implementazione è Commander.js[8] che permette di definire i flag e i comandi che

[8] `https://github.com/tj/commander.js/`

possono essere utilizzati da CLI e inoltre permette di gestire gli argomenti passati da linea di comando in modo semplice e veloce. Segue un esempio di codice che mostra la definizione di un comando, con azione associata, e di un flag.

```
import akomandoDb from 'commander';
akomandoDb
        .version('0.0.1')
        .command('push [par...]')
        .action(function (par) {
                ...
        }
akomandoDb
        .option('--all','Use this flag with pull manifestation
           to list all manifestations in the DB', '' )
```

API akomando-db. Questo modulo è il punto di accesso alle funzionalità di *akomando-db* da parte degli utilizzatori. Oltre a definire le API e inoltrare le richieste ai modulo sottostanti, questo modulo è responsabile di diverse attività:

- definisce i controlli relativi ai parametri passati dall'utente, chiamando la callback con un errore nel caso in cui vi siano parametri non accettati;
- istanzia la classe relativa al database che l'utente vuole utilizzare (backend_exist o backend_fs), passando tutti i parametri per l'inizializzazione del db;
- istanzia, a secondo dell'operazione richiesta dall'utente (push o pull), la classe "akomando-db_push" o "akomando-db_pull" passando i parametri definiti dall'utente e l'istanza del db che poi sarà utilizzata per eseguire concretamente l'azione richiesta.

Ad esempio, un controllo che viene eseguito in questo modulo è la verifica del file passato in input per accertare che questo sia un *Akoma Ntoso* oppure un'altro tipo di file. Tale controllo è fatto per evitare che un utente possa caricare file nel database, come manifestation, che non siano *Akoma Ntoso*. Segue il controllo effettuato tramite RegExp:

```
var is_akomantoso = el.match(/^(<\?xml.*?(\?>)\s*<(\S+:)?
    akomaNtoso|\s*<(\S+:)?akomaNtoso)/g)
if (!(is_akomantoso)){
        return(callback(/*error*/))
}
```

Una volta eseguiti tutti i controlli e create le due istanze, viene chiamato il modulo "akomando-db_switcher" passando i parametri e l'istanza adatta all'esecuzione dell'operazione richiesta.

akomando-db_switcher. Questo modulo, come suggerito dal nome, serve per individuare la specifica funzione da svolgere a partire dai parametri impostati dall'utente. akomando-db_switcher presenta due funzioni "ps" e "pl", entrambe eseguono un insieme di controlli tra i parametri passati per individuare e chiamare la funzione adatta a soddisfare le richieste dell'utente. Quest'ultima sarà un metodo della classe istanziata nel modello trattato precedentemente e passata come argomento ad akomando-db_switcher.

akomando-db_push. Questo modulo insieme ad akomando-db_pull rappresenta il cuore di *akomando-db* e propone un'insieme di funzioni che mirano ad archiviare informazioni in un database in modo asincrono. Tale modulo è rappresentato da una classe che, come visto in precedenza, è istanziata dalle API e contiene due campi: un oggetto contenente dei parametri definiti dall'utente e passati in fase di istanziazione (es. zip); e l'istanza del database da utilizzare (fs o exist). Un aspetto interessante di questo modulo è il lavoro concatenato dei metodi che definisce, infatti se l'utente chiede l'archiviazione di una expression verrà richiamata anche l'archiviazione del work relativo all'expression. Questo lavoro è necessario per costruire una certa struttura nel database che permetta ad *akomando-db* di riconoscere, ogni qual volta è necessario, se l'FRBR che sta trattando è un work, un'expression o una manifestation. Tale meccanismo è applicato in modo semplice, scrivendo nel database un file che descrive il ruolo della locazione in cui si trova. Ad esempio:

```
### STATO DB: ./db
### azione 1
```

```
var akomandoDb = new AkomandoDB()
akomandoDb.setDB(config_file_fs)
akomandoDb.push({par:"/akn/it/doc/2017-09-18/", push_type:"work
    "}, () => {})
STATO DB: ./db/akn/it/doc/2017-09-18/.workInit

### azione 2
akomandoDb.push({par:"/akn/it/doc/2017-09-18/ita@/", push_type:"
    expression"}, () => {})
STATO DB: ./db/akn/it/doc/2017-09-18/.workInit
          ./db/akn/it/doc/2017-09-18/ita@/.expressionInit
```

Per implementare la gestine dei file zip è stata utilizzata la libreria adm-zip[9] che permette in modo efficace di accedere a tutti i file contenuti tramite la funzione "getEntries".

In questo modulo viene utilizzata la funzione "getFRBR" della libreria *akomando*, discussa di seguito, per accedere alle informazioni dell'expression e della manifestation relative al documento *Akoma Ntoso*, come di seguito.

```
import { createAkomando } from 'akomando'
const akomando = createAkomando({aknString: file});
var node = myAkomando.getFRBR()
...
var expressionPath = node.expression.uri;
var manifestation = node.manifestation.this;
```

akomando. Nella figura 6.1 è presente un modulo che rappresenta la libreria *akomando*, descritta nella sezione 5.3, in quanto implementa la funzione "getFRBR" che è stata proposta e sviluppata ai fini del mio progetto di tesi. Tale funzione ritorna le informazioni relative alle tre entità FRBR di un dato documento *Akoma Ntoso*. Segue una parte dell'implementazione "getFRBR".

```
let result = {}
let doctype = this.getDocType(); //funzione interna ad akomando
// di seguito acquisisco le informazioni sull'entita' work
let elemWorkThis = this.getElementsFromXPath(`/akomaNtoso/${
    doctype}/meta/identification/FRBRWork/FRBRthis`, true);
```

[9] `https://www.npmjs.com/package/adm-zip`

```
let elemWork = this.getElementsFromXPath(`/akomaNtoso/${doctype
    }/meta/identification/FRBRWork/FRBRuri`, true);
if(elemWork.length > 0 && elemWorkThis.length > 0){
    result.work = {}
    result.work.uri = elemWork[0].getAttribute('value');
    result.work.this = elemWorkThis[0].getAttribute('value');
}
```

akomando-db_pull. akomando-db_pull, come il modulo akomando-db_push, rappresenta il cuore di *akomando-db*. Questo modulo concerne tutte le operazioni che mirano a restituire informazioni all'utente a partire dalle informazioni contenute nel database. Tali operazioni permettono di recuperare dal database:

- singoli documenti *Akoma Ntoso*(pullManifestation);
- singoli allegati (pullAttachment);
- file zip contenenti tutti i file presenti all'interno di un certo FRBR work o FRBR expression (pull work/expression, zip:true);
- liste di tutti i file presenti all'interno di un certo FRBR work o FRBR expression (pull work/expression, zip:false).

Tale modulo è rappresentato da una classe che, come visto in precedenza, è istanziata dalle API e contiene più campi: un oggetto contenente dei parametri definiti dall'utente e passati in fase di istanziazione; l'istanza del database da utilizzare (fs o exist); un campo "zip" per la gestione interna della creazione degli zip risultanti. La libreria adm-zip[10] è stata utilizzata per manipolare i file zip e ha permesso in una prima versione di *akomando-db*, una gestione locale: i file venivano salvati nel file system per poi essere aggiunti al file zip risultante. Successive modifiche e migliorie hanno portato ad utilizzare funzioni della libreria che hanno permesso la creazione dei file zip senza passare dal file system locale, aspetto necessario per la crezione di una libreria con accesso tramite API.

[10] `https://www.npmjs.com/package/adm-zip`

interfaccia DB. L'interfaccia con i database appare unica, nella figura che rappresenta l'architettura in quanto viene definita all'inizio dell'iterazione tra utente e libreria nel momento in cui viene istanziata la classe dell'opportuno database. I nomi delle funzioni che sono mostrate nella figura sono implementati in entrambe le classi così da non avere problemi nelle chiamate fatte da parte dei moduli akomando-db_pull e akomando-db_push; le quali dunque chiameranno sempre la stessa funzione ma cambierà, di volta in volta, l'interprete di quest'ultima. Per quanto riguarda la gestione dell'interfacciamento vero e proprio ai due database supportati sono state utilizzate le librerie fs[11], http[12] e il linguaggio XQuery[13]. La prima libreria è necessaria per gestire il salvataggio e il caricamento dei file dal file system locale; la seconda libreria insieme alle query XQuery è alla base dell'interfaccia per eXist-db che viene fatta tramite delle get e delle put con l'entry point del database. Per informazioni relative ai parameti necessari all'inizializzazione dei database in *akomando-db* si rimanda alla sezione 6.3.

6.5 Dashboard e altri contesti applicati per akomando-db

In questo capitolo è stato descritto *akomando-db*, gli obiettivi che hanno portato al suo sviluppo, alcuni casi d'uso che ne caratterizzano l'utilità e infine le API e la sua architettura. Riprendendo la dashboard, *akomando-db* si posiziona tra i database e il lato server della struttura di *SOFIA* (capitolo 5); per chiarire il ruolo di *akomando-db* in *SOFIA* è riportata l'astrazione mostrata nella figura 6.2 che riprende e raffina la figura mostrata nel capitolo precedente.

Inoltre, come si evince dalla sezione 6.1, *akomando-db* però va oltre il suo utilizzo all'interno di *SOFIA*, esso infatti può essere utilizzato in altri ambienti e all'interno di altri strumenti con obiettivi potenzialmente diversi che condividono lo stesso standard *Akoma Ntoso*. Infatti nei casi d'uso trattati

[11] `https://nodejs.org/api/fs.html`
[12] `https://nodejs.org/api/http.html`
[13] `https://en.wikipedia.org/wiki/XQuery`

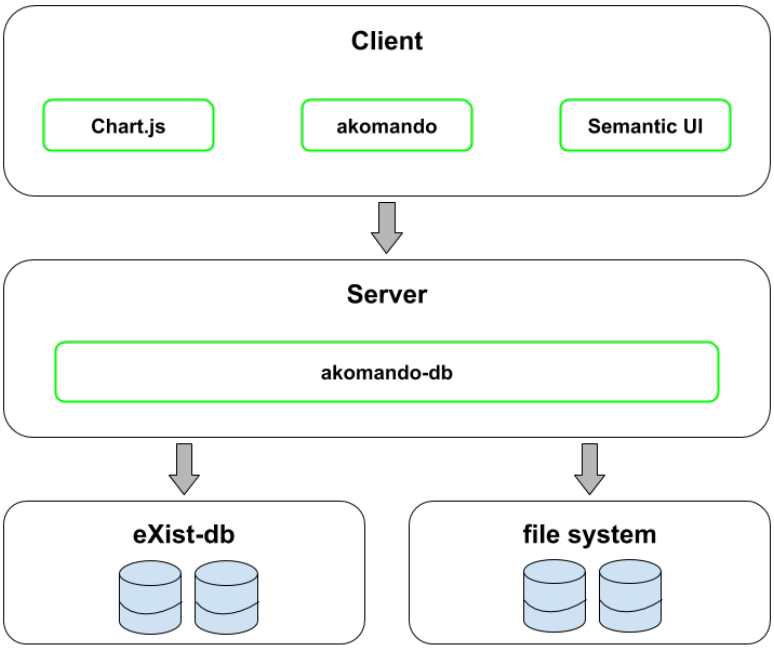

Figura 6.2: Dashboard: astrazione dei moduli principali

nella sezione 6.2 si è parlato di diversi utilizzatori come: il data journalist, utente principale per i fini della tesi di questa dissertazione; il professore che, anche con poca esperienza nell'ambito dell'informatica applicata, può utilizzare dei semplici comandi da CLI per scaricare i documenti *Akoma Ntoso*; il programmatore che sviluppa strumenti per l'Informatica Giuridica che utilizza *akomando-db* come supporto per le fasi di sviluppo e di testing.

Akomando-db è quindi un tool estremamente dinamico, semplice e flessibile, che può essere utilizzato da alcune categorie di utente in ambienti differenti.

Nel prossimo capitolo saranno presentati i risultati emersi dalle analisi descritte nella sezione 5.1, tramite le relative visualizzazioni realizzate con *SOFIA*; infine sarà valutata la tesi descritta nell'introduzione alla luce della dashboard sviluppata, della libreria *akomando-db* e del caso d'uso principale descritto nella sezione 6.2.

Capitolo 7

Risultati e valutazioni

Nei due capitoli precedenti è stata descritta la dashboard sviluppata per i fini di questa dissertazione e le componenti utili per la sua creazione come ad esempio *akomando-db*.

Di seguito, avendo parlato di tutte le componenti che caratterizzano la struttura della dashboard, saranno presentati i risultati ottenuti dalle analisi descritte nella sezione 5.1, che sono state sviluppate e rappresentate graficamente tramite tale dashboard. Infine sarà presentata una valutazione che ha l'obiettivo di verificare che la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

7.1 Risultati della Dashboard

Per lo svolgimento delle analisi, descritte nella sezione "Obiettivi" del capitolo dove è stata presentata la dashboard (5.1), è stato utilizzato un insieme di documenti *Akoma Ntoso*. Quest'ultimi sono stati selezionati in modo casuale tra i documenti legislativi italiani che vanno dal 1990 al 2017 e sono ottenuti tramite un processo automatico[1] all'interno del CIRSFID[2]. Il numero totale di documenti analizzati è 3138 ed essi si dividono in tre tipologie: leggi, decreti legge, decreti legislativi.

Seguono i risultati delle analisi divisi in due sezioni: analisi degli articoli e analisi dei paragrafi.

7.1.1 Analisi basate sugli articoli

Questa tipologia di analisi tratta il numero di articoli che compongono un documento legislativo al fine di osservare se e come l'organizzazione strutturale dei documenti cambia nel tempo. Tali analisi sono affiancate da grafici che mostrano la lunghezza, intesa come numero di caratteri, relativa al contenuto del corpo dei documenti legislativi.

La figura 7.1 mostra il numero medio di articoli (asse y) presenti nei documenti legislativi raggruppati per anno; la figura 7.2 presenta il numero medio di caratteri contenuti nella parte centrale dei documenti raggruppati per anno.

[1] L'insieme di documenti utilizzati per le analisi non contengono gli allegati del corpo legislativo bensì solo gli articolati

[2] `http://www.cirsfid.unibo.it/`

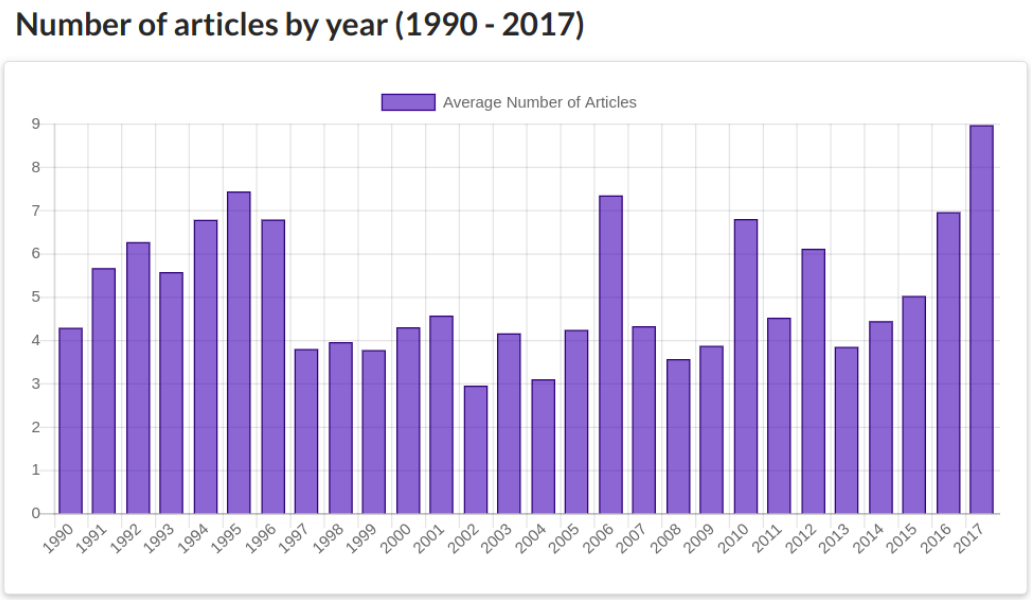

Figura 7.1: Analisi sugli articoli (per anno)

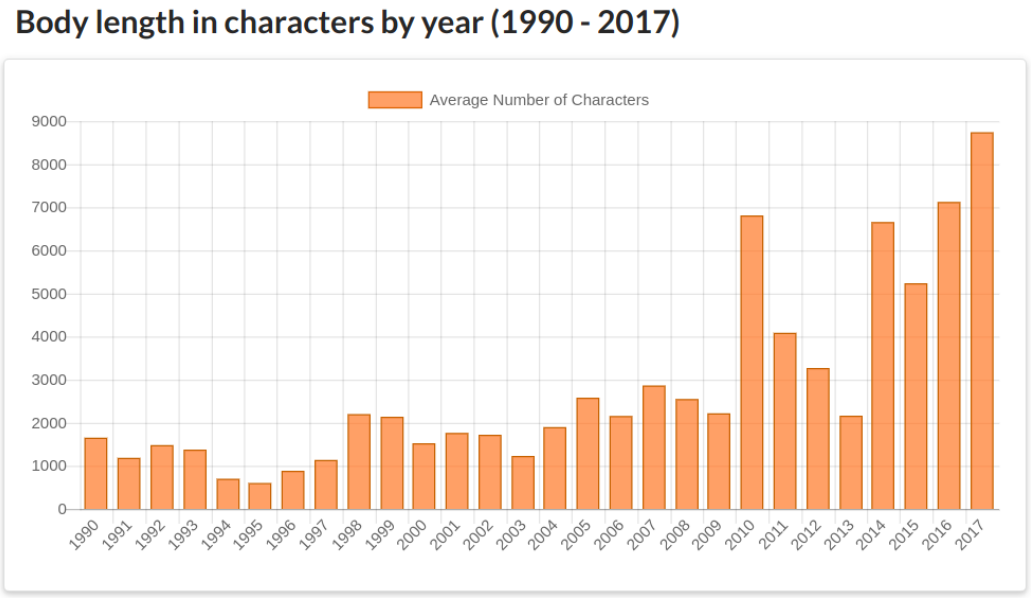

Figura 7.2: Analisi della lunghezza dei documenti (per anno)

In modo simile ma da una prospettiva diversa, la figura 7.3 mostra il numero medio di articoli (asse y) presenti nei documenti raggruppati per

legislatura; la figura 7.4 mostra il numero medio di caratteri contenuti nella parte centrale dei documenti raggruppati per legislatura.

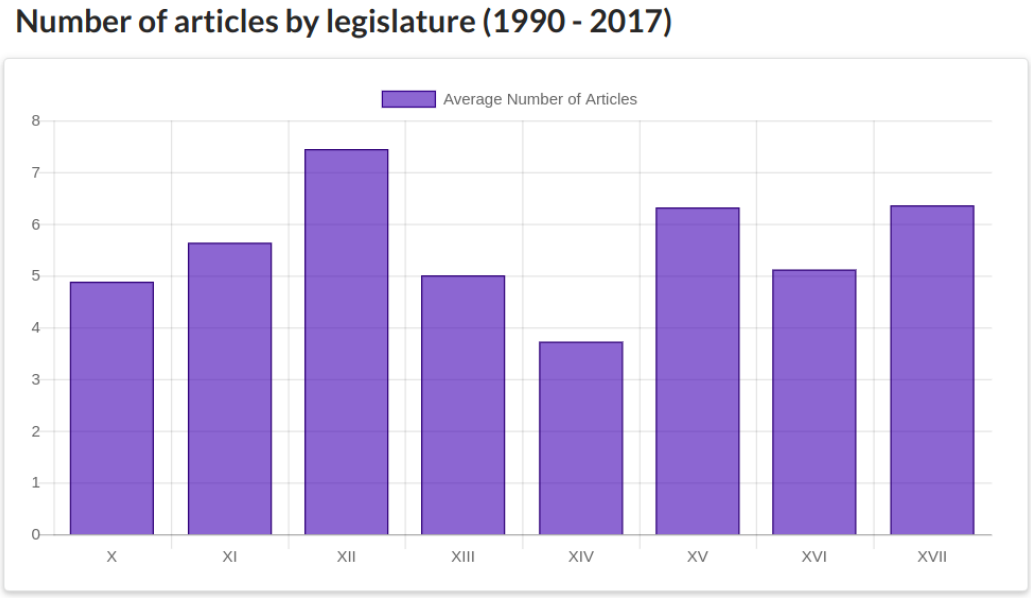

Figura 7.3: Analisi sugli articoli (per legislatura)

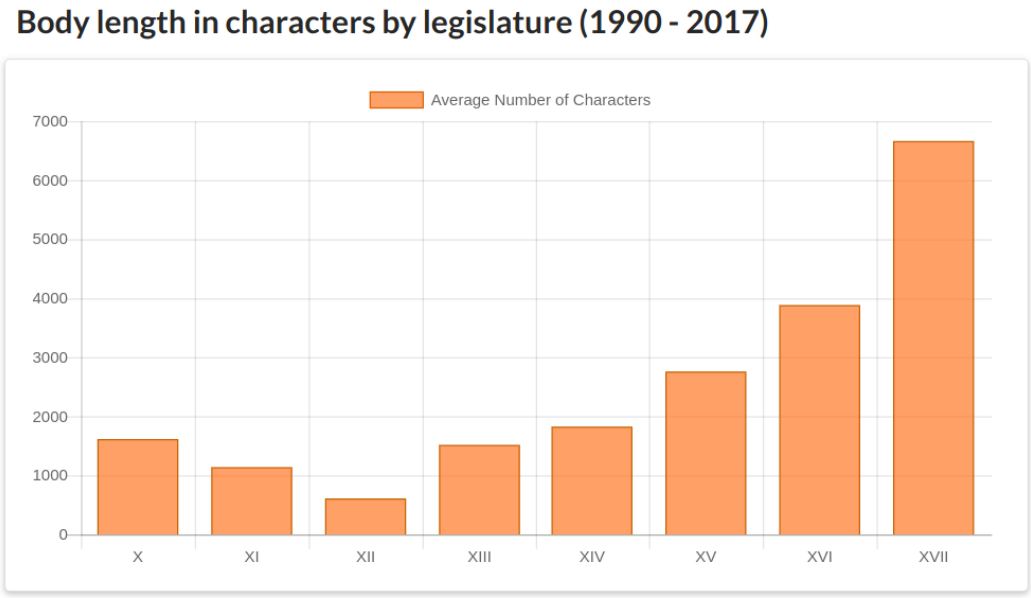

Figura 7.4: Analisi della lunghezza dei documenti (per legislatura)

7.1.2 Analisi basate sui paragrafi

Questa tipologia di analisi ha lo stesso obiettivo della precedente ma, in modo complementare, cerca di far emergere informazioni relativi ai paragrafi che compongono un documento legislativo. In tal modo è possibile avere un quadro più completo sull'organizzazione strutturale dei documenti, dipendentemente dal periodo storico e dalle differenti legislature. Come in precedenza, le analisi mostrate di seguito sono affiancate da grafici che mostrano la lunghezza, intesa come numero di caratteri, relativa al contenuto della parte centrale dei documenti legislativi.

La figura 7.5 mostra il numero medio dei paragrafi (asse y) presenti nei documenti legislativi raggruppati per anno; la figura 7.6 presenta il numero medio di caratteri relativi ai documenti raggruppati per anno e viene riproposta per favorire la comprensione e la comparazione dei grafici.

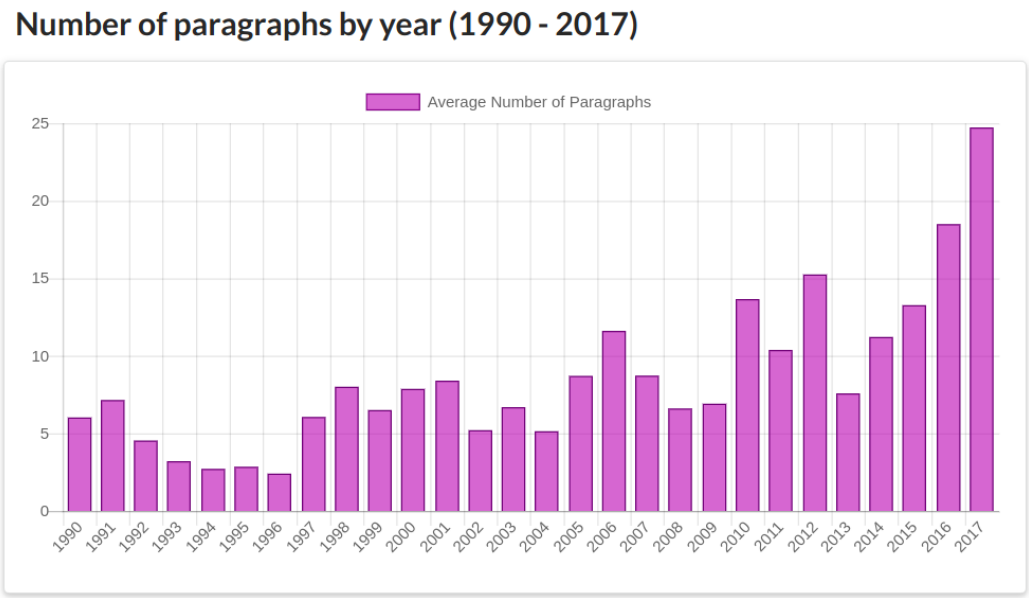

Figura 7.5: Analisi sui paragrafi (per anno)

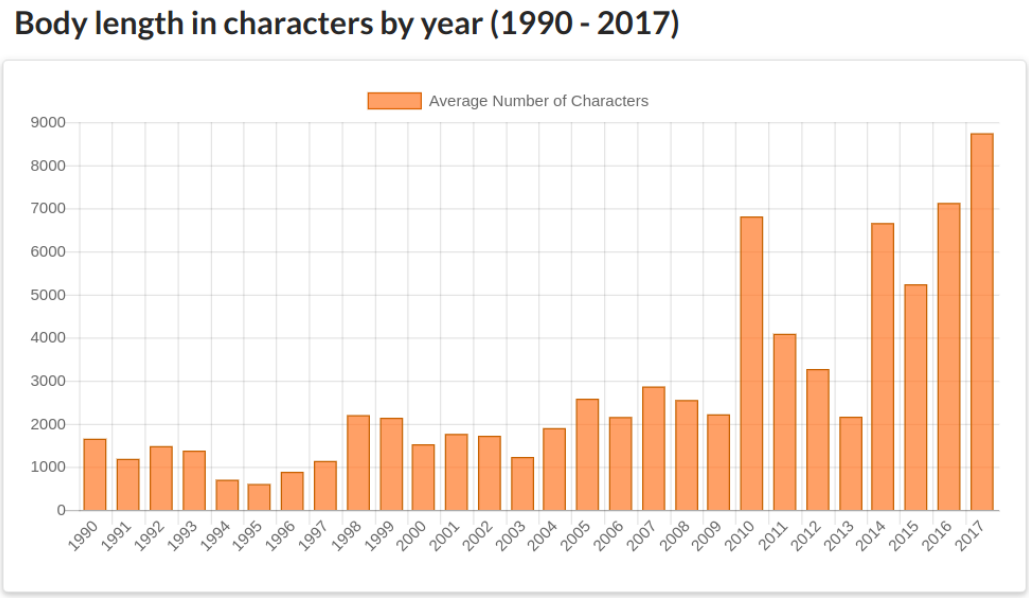

Figura 7.6: Analisi della lunghezza dei documenti (per anno)

Dal un altro punto di vista, la figura 7.7 mostra il numero medio dei paragrafi (asse y) relativo ai documenti raggruppati per legislatura; la figura

7.8 è relativa alla lunghezza dei documenti e viene riproposta per favorire la comprensione e la comparazione dei grafici.

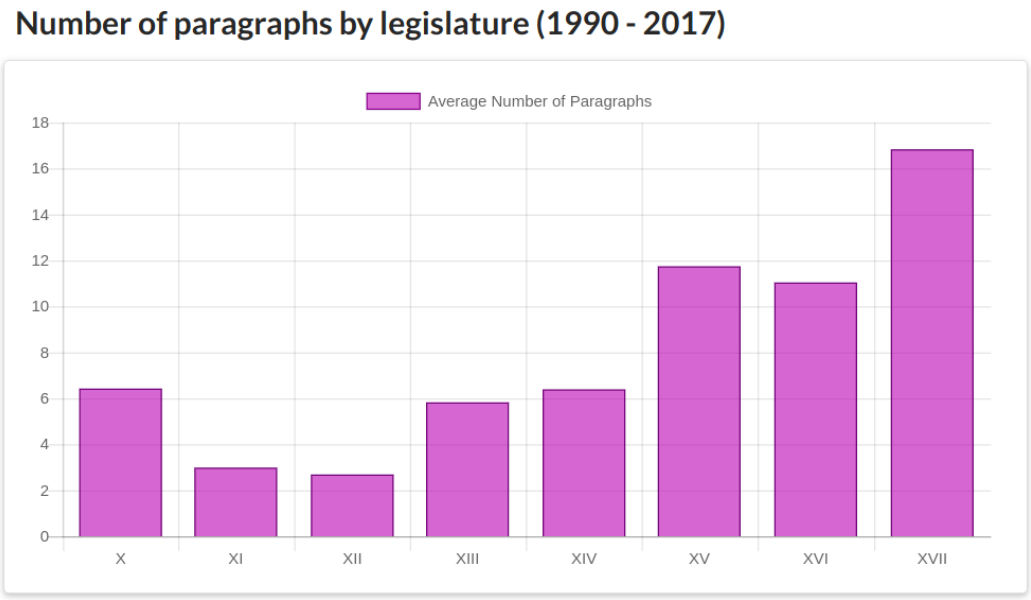

Figura 7.7: Analisi sui paragrafi (per legislatura)

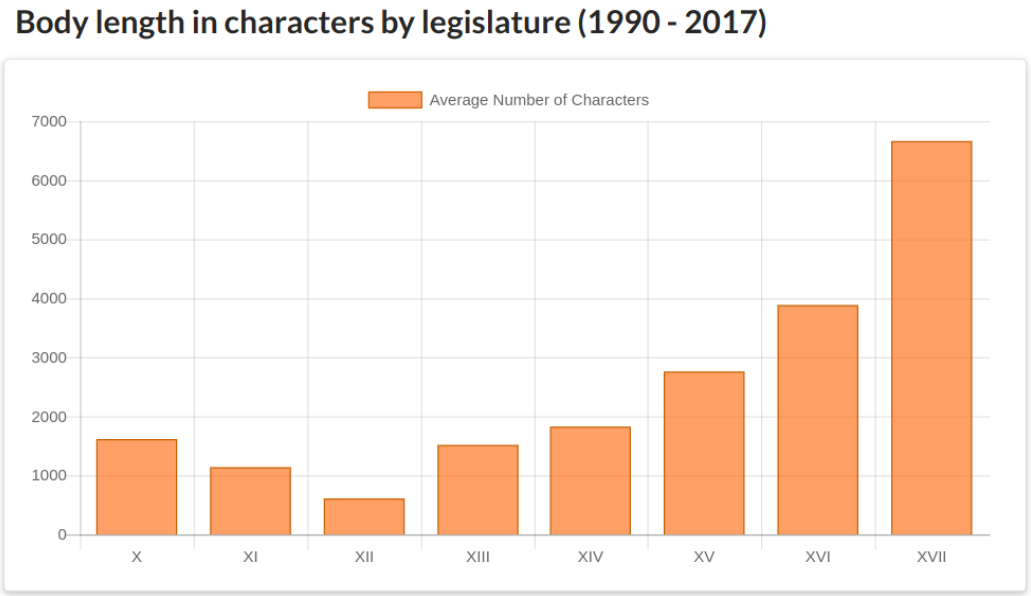

Figura 7.8: Analisi della lunghezza dei documenti (per legislatura)

7.1.3 Analisi basata sulle tipologie di documento legislativo

Diversamente dalle precedenti, l'analisi che segue non tratta i vari documenti indistintamente, ha invece l'obiettivo di far emergere le potenziali differenze tra le tre tipologie di documento giuridico che popolano il database di riferimento: decreto legislativo, decreto legge e legge. L'idea è quella di analizzare l'organizzazione strutturale dei documenti appartenenti a queste tre diverse tipologie, relativamente al numero di articoli, di paragrafi, e alla lunghezza della loro parte centrale. Le analisi che si basano su questi tre criteri, presentate di seguito, sono eseguite su documenti raggruppati per legislatura.

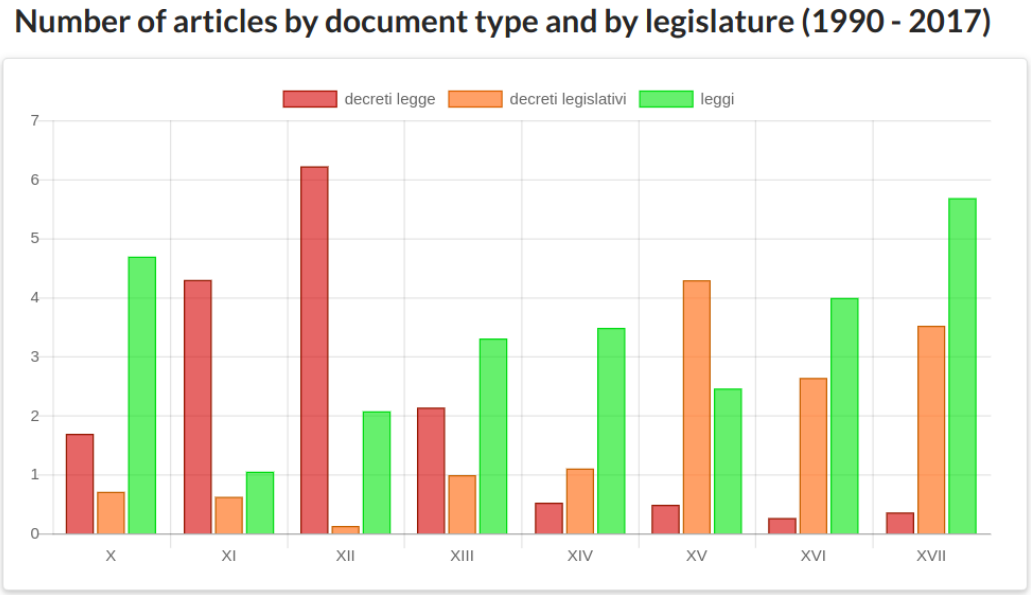

Figura 7.9: Analisi sul numero di articoli delle tre tipologie di documento

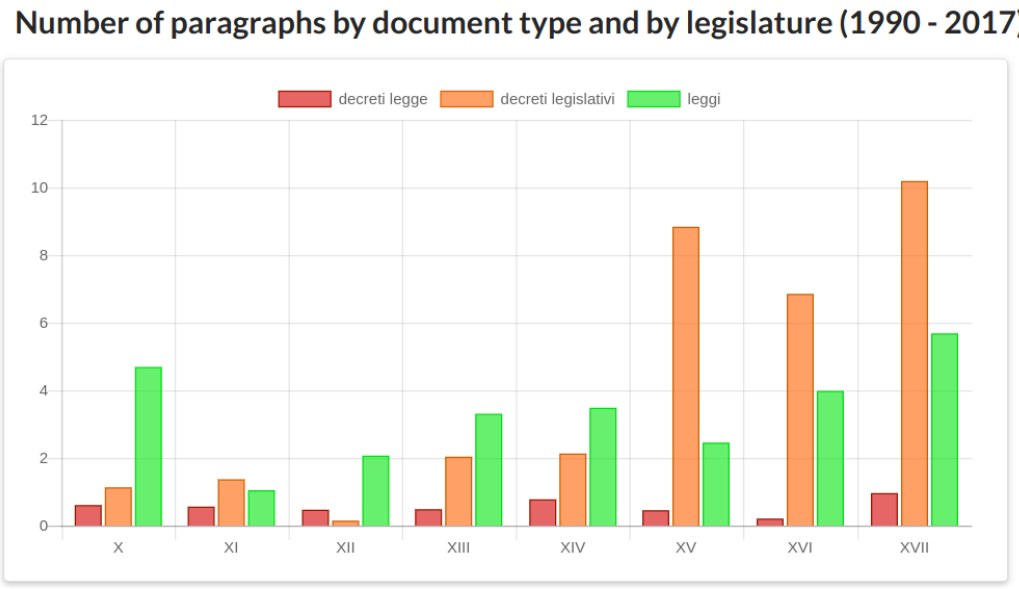

Figura 7.10: Analisi sul numero di paragrafi delle tre tipologie di documento

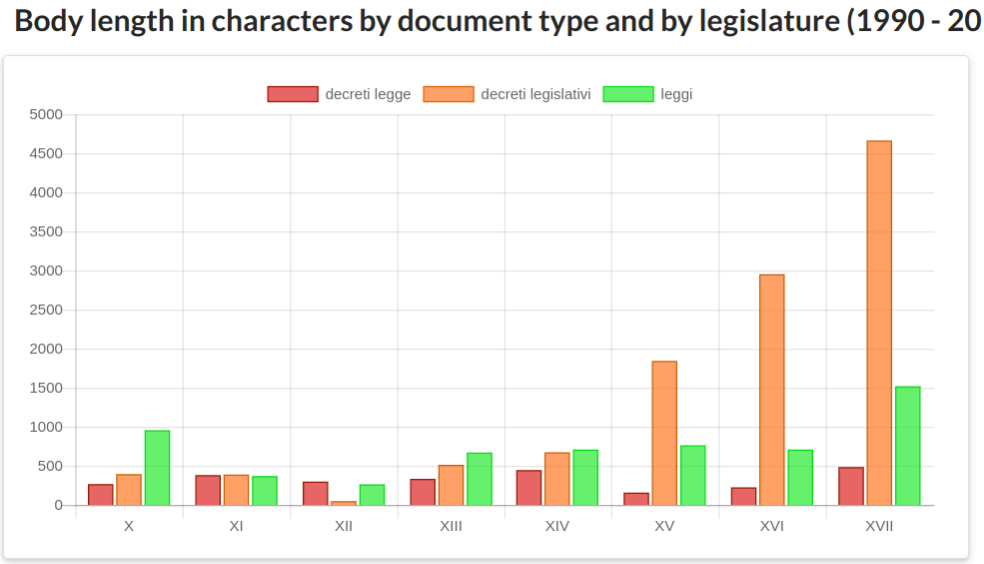

Figura 7.11: Analisi sulla lunghezza del testo delle tre tipologie di documento

7.1.4 Osservazioni sui risultati

Osservando le rappresentazioni grafiche si può notare un aumento costante del numero di paragrafi e di caratteri nei documenti legislativi a partire dal-

l'anno 1990 fino al 2017. In particolare le ultime tre legislature sono quelle in cui si evidenzia un incremento netto del numero medio di paragrafi e di caratteri. Spostando l'attenzione sui grafici relativi al numero di articoli si evidenzia invece un andamento quasi inverso a quello descritto per i paragrafi ed i caratteri; infatti dal 1997 il numero medio di articoli per documento rimane sostanzialmente più basso del periodo precedente per poi iniziare nuovamente a crescere dal 2014. Infine, osservando i grafici in cui vengono messi a confronto i diversi tipi di documento legislativo, si nota che l'aumento del numero medio di caratteri e di paragrafi è associato principalmente ai decreti legislativi.

7.2 Valutazione finale

Questa dissertazione ha trattato indipendentemente due importanti discipline, l'Informatica Giuridica e la Data Analysis, per poi metterle insieme al fine di ricercare cosa già è stato fatto e far emergere le potenzialità nate dalla loro unione. Dopo aver trattato questi argomenti si è passati alla parte implementati, mostrando un esempio di Data Journalism dove ho definito alcune analisi su documenti *Akoma Ntoso* e ho sviluppato *SOFIA*, uno strumento che permettesse di comunicare i risultati. Durante tale discussione è emersa la mancanza di uno strumento in grado di lavorare e interfacciarsi in modo semplice con i vari database; questa mancanza mi ha portato a sviluppare e presentare il cuore del mio progetto di tesi, *akomando-db*. Con il supporto di questo strumento è stata completata la discussione riguardante lo sviluppo di *SOFIA*.

In questo capitolo sono stati discussi i risultati ottenuti insieme ad alcune osservazioni sui grafici ma l'interesse ultimo è verificare se la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

Per arrivare alla formulazione di una risposta a questa tesi ho deciso di iniziare proprio dai risultati ottenuti che rappresentano sia un esempio di analisi possibili con i documenti *Akoma Ntoso*, sia un esempio dello sviluppo di strumenti per andare in direzione degli obiettivi della Data Visualization.

Segue una valutazione qualitativa dove sarà messo a confronto il lavoro svolto per la creazione della dashboard, visto nei capitoli 5 e 6, con il potenziale carico di lavoro necessario per il suo sviluppo, in Node.js, senza l'utilizzo di meccanismi che semplifichino la gestione dell'archiviazione e del recupero dei documenti *Akoma Ntoso*. Il caso d'uso di riferimento è quello di Sabrina, una data journalist di 40 anni con un background informatico (sezione 6.2).

Dashboard senza l'ausilio di akomando-db. Ipotizzando di non avere a disposizione uno strumento come *akomando-db* che porta con sé una serie di vantaggi agli utilizzatore, trattati dettagliatamente nella sezione 6.1, si cercherà di descrivere in modo discorsivo le tecnologie e i concetti che devono essere utilizzati per la gestione delle fasi di archiviazione e recupero per i documenti *Akoma Ntoso*.

Innanzitutto, a livello concettuale, il data journalist deve conoscere la naming convention dello standard *Akoma Ntoso*, usata per l'identificazione univoca delle risorse, al fine di poter utilizzare lo stesso meccanismo per il salvataggio dei documenti e il successivo recupero, ponendolo in un contesto dov'è specificato un certo database. Tale situazione richiede che l'utente conosca anche il modello concettuale FRBR e le diverse entità: work, expression, manifestation; tale conoscenza, unita alla naming convention, permetterà all'utente di comprendere gli identificatori univoci discussi nella sezione 4.3.3. Le competenze richieste all'utente sono direttamente collegate alle varie tipologie di database che si vogliono gestire poiché ognuna di queste presenta meccanismi e strumenti specifici che ne permettono l'utilizzo in modo corretto e controllato. In modo corretto significa che ogni richiesta definisce un'azione ben precisa che deve essere soddisfatta in modo appropriato; con il termine controllato si intende che l'utilizzatore deve essere informato di eventuali problemi o errori connessi al modo di utilizzo oppure a fattori esterni. Quindi il data journalist deve conoscere i vari database da gestire; analizziamo i due casi presi in considerazione nella dashboard:

- **file system**: in questo caso è necessario conoscere una libreria, come fs[3], per gestire un database in un file system. E' necessario implemen-

[3] `https://nodejs.org/api/fs.html`

tare la ricerca di documenti e il salvataggio di documenti nel modo corretto e coerente con la naming convention al fine di garantire una certa struttura semantica al database e non incorrere in problemi di univocità;

- **eXist-db**: per utilizzare il database eXist-db è necessario conoscere i meccanismi interni e gli strumenti che mette a disposizione per richiedere o caricare risorse. Innanzitutto si deve conoscere il modo con cui collegarsi al database eXist, gestendo l'endpoint e altre informazioni come la porta e le credenziali d'accesso. Inoltre è necessario conoscere e saper costruire delle query tramite XQuery[4], linguaggio per la ricerca di documenti e l'estrazione di elementi e attributi da documenti XML, in modo da richiedere le giuste informazioni, formattate in modo opportuno. Per la creazione di query XQuery corrette è necessario conoscere e saper utilizzare i namespace di cui si necessita.

Per la gestione dei file zip e quindi per poter caricare tutti i documenti contenuti in un file zip è necessario utilizzare una libreria come adm-zip[5] per estrapolare e utilizzare i documenti contenuti all'interno di un file zip. Infine un altro aspetto che necessita di conoscere specifici meccanismi e controlli è quello collegato alla sovrascrittura o meno dei documenti già presenti in un database.

Dashboard con l'ausilio di akomando-db. In quest'altro caso, il data journalist dovrà solamente leggere la documentazione e fare un mapping delle sue necessità con le API fornite da *akomando-db*. Non necessita di conoscere la naming convention di *Akoma Ntoso*, nè il modello concettuale FRBR. Non dovrà sentir parlare di librerie come fs e neanche del linguaggio XQuery. Semplicemente il data journalist potrebbe beneficiare del codice di cui si compone *akomando-db* attraverso le funzioni di API che vengono esposte agli utilizzatori.

[4] `https://www.w3schools.com/xml/xquery_intro.asp`
[5] `https://www.npmjs.com/package/adm-zip`

Alla luce delle tecnologie e dei meccanismi che è necessario conoscere e alla complessità che presentano per essere utilizzate è possibile affermare che la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi per le analisi e facilita la creazione di strumenti per le visualizzazioni dei risultati.

La valutazione che mi ha permesso di confermare la tesi si basa quindi sul carico di studio delle tecnologie e dei meccanismi che un data journalist deve sostenere per la costruzione di uno strumento di Data Visualization con certe caratteristiche e per la realizzazione di analisi su documenti *Akoma Ntoso* attraverso una libreria come *akomando*.

Capitolo 8

Conclusioni

Questa dissertazione è iniziata con un'affermazione che ha influenzato e indirizzato questo lavoro; tale tesi afferma che la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

L'obiettivo finale, quindi, è stato di accettare o confutare l'affermazione iniziale e giustificare tale risposta con delle motivazioni solide e ammissibili. Questo obiettivo è stato raggiunto in modo costruttivo, sviluppando un percorso che ha inizialmente fornito i concetti base al lettore per la comprensione del contesto in cui si concretizza il lavoro svolto; in seguito si è introdotta la parte implementativa del progetto presentando *akomando-db* e *SOFIA*. Questi due strumenti hanno poi permesso di arrivare a risultati interessanti, discussi nella sezione 7.1, e alla valutazione finale della tesi, discussa nella sezione 7.2.

Inizialmente si è discusso del ruolo dei dati digitali e di come questi possono essere manipolati attraverso un processo di analisi per l'estrapolazione di informazioni, le quali in seguito possono essere comunicate al destinatario tramite meccanismi di Data Visualization. E' stato inoltre introdotto l'ambito del Data Journalism che risulta importante per la comprensione del caso d'uso utilizzato nella valutazione finale (sezione 7.2); infatti ho individuato nel data journalist l'utente ideale, per competenze e obiettivi, allo sviluppo di uno strumento come *SOFIA* tramite *akomando-db*. Successivamente è sta-

ta introdotta la disciplina dell'Informatica Giuridica per arrivare a discutere dell'ambito relativo alla standardizzazione dei documenti giuridici in formato digitale e a presentare gli strumenti necessari a questo settore, ovvero i documenti strutturati, il linguaggio XML e i relativi standard come ad esempio *Akoma Ntoso*. Avendo introdotto questi strumenti si è passati alla presentazione di alcuni esempi che mettono insieme l'ambito della Data Analysis e della Data Visualization con l'ambito dell'Informatica Giuridica al fine di mostrare alcuni risultati raggiunti e le potenzialità di tale cooperazione.

Successivamente si è passati alla parte implementativa presentando due strumenti: la dashboard *SOFIA* (Structured Open government data For Interactive Analysis) e la libreria *akomando-db*. *SOFIA* è stata ideata come un esempio di strumento per l'esecuzione di analisi su collezioni di documenti e per la presentazione delle relative rappresentazioni grafiche. Il ruolo di *SOFIA* è anche quello di mettere in evidenza le potenzialità che nascono dall'analisi di documenti giuridici come mostrato dalle rappresentazioni grafiche riportate nella sezione 7.1. *Akomando-db* invece è il cuore del mio progetto di tesi che offre un insieme di funzionalità, descritte in dettaglio nella sezione 6.1, per supportare il lavoro con documenti *Akoma Ntoso*. *Akomando-db* risulta fondamentale ai fini della dissertazione perché permette di confermare la tesi iniziale in quanto semplifica in modo significativo il lavoro svolto per lo sviluppo di *SOFIA*. Oltre al suo contributo all'interno di questa dissertazione, *akomando-db* vuole essere un contributo scientifico allo standard *Akoma Ntoso* semplificando il suo utilizzo e supportando la sua diffusione. La libreria *akomando-db* è stata inoltre sottoposta alla community di *akomando* al fine di essere rilasciata su npm.

Si è arrivati alla valutazione finale quindi utilizzando *SOFIA* e ipotizzando due differenti scenari per il suo sviluppo: uno in cui il data journalist utilizza *akomando-db*; l'altro in cui viene discusso lo sviluppo di *SOFIA* senza l'ausilio di *akomando-db*. Il primo caso si concretizza con il lavoro presentato nella dissertazione in cui viene descritta la struttura di *SOFIA* utilizzando la libreria sviluppata. Il secondo caso, invece, ha fatto emergere una moltitudine di tecnologie e concetti, alcuni abbastanza complessi, che un data journalist deve conoscere al fine di sviluppare le varie componenti che costituiscono la struttura di *SOFIA*. Alla luce della complessità e del tempo

necessario alla comprensione e all'utilizzo dei vari strumenti, si è arrivati alla conferma della tesi secondo cui la semplificazione delle attività di archiviazione e recupero dei documenti giuridici nello standard più appropriato supporta meccanismi di Data Analysis e favorisce lo sviluppo di strumenti per la Data Visualization.

Akomando-db presenta diversi margini di miglioramento e di espansione delle sue funzionalità in quanto è uno strumento abbastanza giovane.

Si ipotizzi di voler espandere il lavoro svolto con *SOFIA* e quindi, di voler sviluppare analisi più complete che prendono in considerazione anche gli allegati associati ai vari documenti giuridici. In un contesto del genere è necessario avere uno strumento utile alla gestione completa, efficace e corretta degli allegati relativi ai vari documenti *Akoma Ntoso*. Un primo approccio a questo nuovo obiettivo potrebbe essere l'espansione e l'arricchimento di *akomando-db* al fine raffinare e completare la gestione degli allegati; tale gestione non è semplice in quanto gli allegati sono fortemente dipendenti dai documenti a cui sono associati, è quindi necessario manipolare la fase di archiviazione per preservare tali associazioni.

Andando in una direzione differente rispetto alla precedente, ci si concentri sull'utilizzatore finale e sul miglioramento dell'interazione con la libreria *akomando-db* in modo tale da renderne più semplice e gratificante l'utilizzo. In questo contesto reputo interessante l'espansione di *akomando-db* con funzionalità per il controllo del flusso di lavoro inteso come l'insieme di operazioni svolte da *akomando-db* dal momento in cui l'utente ne richiede l'utilizzo al momento in cui l'intera operazione termina. L'idea è quella di permette all'utilizzatore di definire delle regole tramite ad esempio una callback per monitorare in modo continuo le attività svolte dalla libreria ed eventualmente anticiparne la terminazione se una certa condizione si verifica.

Un ultimo scenario possibile per gli sviluppi futuri di *akomando-db* potrebbe nascere dalla necessità di avere un unico strumento che metta insieme il ruolo di *akomando* e quello di *akomando-db*. Quindi, l'idea finale è quella di sviluppare una libreria che riesca a gestire le fasi di archiviazione e di recupero dei documenti e che permetta anche di eseguire analisi su questi in modo da restituire direttamente i risultati ottenuti. Tale scenario rappresenta una vera e propria evoluzione di *akomando-db* che possiede già le basi per questa

direzione ma che necessita di molto lavoro per mantenere l'indipendenza dai database sottostanti andando a creare delle procedure specifiche di estrapolazione dei dati per ogni database che si vuole gestire.

In questo lavoro di tesi ho quindi evidenziato, tramite *SOFIA*, le potenzialità della Data Analysis applicata ai documenti giuridici e, al fine di contribuire a tale ambito, ho sviluppato la libreria *akomando-db* che mira alla diffusione e all'affermazione dello standard *Akoma Ntoso*.

Riferimenti

[AT18] AgID, & Team Digitale. (visited in November 2018). Piano di sviluppo DAF. Data & Analytics Framework Italia. https://docs.italia.it/italia/daf/daf-piano-di-sviluppo/it/bozza/

[BCP10] Barabucci, G., Cervone, L., Palmirani, M., Peroni, S., & Vitali, F. (2010). Multi-layer markup and ontological structures in Akoma Ntoso. In *AI Approaches to the Complexity of Legal Systems. Complex Systems, the Semantic Web, Ontologies, Argumentation, and Dialogue* (pp. 133-149). Springer, Berlin, Heidelberg. 10.1007/978-3-642-16524-5_9.

[Beh97] Behrens, J. T. (1997). Principles and procedures of exploratory data analysis. *Psychological Methods*, 2(2), 131. 10.1037/1082-989X.2.2.131.

[BFP08] Biasiotti, M., Francesconi, E., Palmirani, M., Sartor, G., & Vitali, F. (2008). Legal informatics and management of legislative documents. *Global Center for ICT in Parliament Working Paper*, 2. https://www.researchgate.net/publication/253087975_Legal_Informatics_and_Management_of_Legislative_Documents

[BFS03] Biagioli, C., Francesconi, E., Spinosa, P., & Taddei, M. (2003, June). The NIR project: Standards and tools for legislative drafting and legal document web publication. In *Proceedings of ICAIL workshop on e-government: modelling norms and concepts as key issues* (pp. 69-78). https://www.researchgate.net/publication/228939928_The_NIR_project_Standards_and_tools_for_legislative_drafting_and_legal_document_web_publication

[CRD87] Coombs, J. H., Renear, A. H., & DeRose, S. J. (1987). Markup systems and the future of scholarly text processing. *Communications of the ACM*, 30(11), 933-947.

[DDK15] D'Acquisto, G., Domingo-Ferrer, J., Kikiras, P., Torra, V., de Montjoye, Y.-A., & Bourka, A. (2015, December). Privacy by design in big data (report). European Union Agency for Network and Information Security (ENISA).
https://www.enisa.europa.eu/publications/big-data-protection

[Eco10] Economist, T. (2010, February 25). Data, data everywhere. The Economist.
https://www.economist.com/special-report/2010/02/25/data-data-everywhere

[Few06] Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media, Inc.

[FP16] Faini, F., & Palmirani, M. (2016, September). Italian Open and Big Data Strategy. In *International Conference on Electronic Government and the Information Systems Perspective* (pp. 105-120). Springer, Cham.

[Her18] Heravi, B. (2018). 3WS of Data Journalism Education: What, where and who?. *Journalism Practice*, 1-18. 10.1080/17512786.2018.1463167.

[IFLA98] IFLA Study Group on the Functional Requirements of Bibliographic Records. (1998). Functional Requirements of Bibliographic Records: Final Report. München: K. G. Saur.
https://www.ifla.org/files/assets/cataloguing/frbr/frbr.pdf

[KK11] Khan, M., & Khan, S. S. (2011). Data and Information Visualization Methods, and Interactive Mechanisms: A Survey. *International Journal of Computer Applications*, 34(1), 1-14.
https://www.researchgate.net/publication/264623537_Data_and_Information_Visualization_Methods_and_Interactive_Mechanisms_A_Survey

[LJ12] Labrinidis, A., & Jagadish, H. V. (2012). Challenges and opportunities with big data. *Proceedings of the VLDB Endowment*, 5(12), 2032-2033. 10.14778/2367502.2367572.

[MBB09] Mazzega, P., Bourcier, D., & Boulet, R. (2009). The network of French legal codes. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law (ICAIL 2009)* (pp. 236-237). Barcelona, Spain — June 08-12, 2009. ACM. 10.1145/1568234.1568271

[MMS02] Marchetti, A., Megale, F., Seta, E., & Vitali, F. (2002). Using XML as a means to access legislative documents: Italian and foreign experiences. *ACM SIGAPP Applied Computing Review*, 10(1), 54-62. 10.1145/568235.568246.

[OASIS17] OASIS. (2017). Akoma Ntoso Naming Convention Version 1.0. `http://docs.oasis-open.org/legaldocml/akn-nc/v1.0/akn-nc-v1.0.html`

[OS13] O'Neil, C., & Schutt, R. (2013). *Doing data science: Straight talk from the frontline*. O'Reilly Media, Inc.

[Pal15] Palmirani, M. (2015). Definizione dell'Informatica Giuridica e inquadramento storico. Slide del corso di Informatica Giuridica, a.a. 2014/2015, Università di Bologna. `https://informaticagiuridica4ra.files.wordpress.com/2015/05/lezione01-2015.pdf`

[PBC18] Palmirani, M., Bianchi, I., Cervone, L., & Draicchio, F. (2018). Analysis of Legal References in an Emergency Legislative Setting. In *AI Approaches to the Complexity of Legal Systems (AICOL 2015, AICOL 2016, AICOL 2016, AICOL 2017, AICOL 2017)*. Lecture Notes in Computer Science, vol 10791. Springer, Cham. 10.1007/978-3-030-00178-0_20.

[PMG14] Palmirani, M., Martoni, M., & Girardi, D. (2014, September). Open Government Data Beyond Transparency. In *International Conference on Electronic Government and the Information Systems Perspective (EGOVIS 2014)*. Lecture Notes in Computer Science, vol 8650,

pp. 275-291. Munich, Germany, September 1-3, 2014. Springer, Cham. 10.1007/978-3-319-10178-1_22.

[PV12] Palmirani, M., & Vitali, F. (2012). *Legislative XML: principles and technical tools*. Inter-American Development Bank.

[RD00] Rahm, E., & Do, H. H. (2000). Data cleaning: Problems and current approaches. *IEEE Data Eng.* Bull. 23(4), 3-13.

[Ric06] Richmond, B. (2006). *Introduction to data analytics handbook*. Migrant and Seasonal Head Start Technical Assistance Center, Academy for Educational Development.
`https://files.eric.ed.gov/fulltext/ED536788.pdf`

[Sar16] Sartor, G. (2016). *L'informatica giuridica e le tecnologie dell'informazione: Corso di informatica giuridica* (Vol. 2). G Giappichelli Editore.

[Tuk62] Tukey, J. W. (1962). The Future of Data Analysis. *The annals of mathematical statistics*, 33(1), 1-67. 10.1214/aoms/1177704711.
`https://projecteuclid.org/euclid.aoms/1177704711`

[Tuk80] Tukey, J. W. (1980, February). We Need Both Exploratory and Confirmatory. *The American Statistician*, 34(1), 23-25. 10.1080/00031305.1980.10482706.
`https://www.jstor.org/stable/2682991`

[Uba13] Ubaldi, B. (2013), "Open Government Data: Towards Empirical Analysis of Open Government Data Initiatives", *OECD Working Papers on Public Governance*, No. 22, OECD Publishing, Paris. 10.1787/5k46bj4f03s7-en.

[UN18] United Nations. (visited in November 2018). Akoma Ntoso.
`http://www.akomantoso.org/`

[W3C18] World Wide Web Consortium. (visited in November 2018). An Introduction to Multilingual Web Addresses.
`https://www.w3.org/International/articles/idn-and-iri/`

"So Long, and Thanks for All the Fish"
Douglas Adams