## File System

Il sistema lavora su un File-System centralizzato (HDFS (?)). L’utente ha la piena libertà di gestire il File-System a sue piacere creando cartelle, rinominando file o sopstandoli da una cartella ad un’altra.

## Primitive

Il sistema offre 2 primitive di sistema all’utente utilizzatore. Queste sono:

* Memorizzazione
* Indicizzazione

## Memorizzazione – caricamento

L’utente seleziona i file da inserire nel File-System dedicato in una cartella da lui creata.Il sistema memorizza nel indice (db A) le meta informazioni relative ai file caricati quali:

* Nome file: il nome del documento salvato
* Path file: il nome del path selezionato per la memorizzazione del documento
* Data caricamento: data di caricamento/modifica del documento
* Formato file: estensione del file (.pdf, .pptx, .ppt) utile per la fase di memorizzazione
* Indexed: valore booleano (false se il documento non ha subito indicizzazione, true altrimenti)

[DA INSERIRE IMG (?)

Parallelamente il sistema elabora il raw file ed estraendo dal documento informazioni relative al testo memorizzandole nel (db B). Nello specifico il sistema estrae per ogni pagina/slide delfile caricato:

* Testo
* Numero pagina

Tale operazione viene effettuata sfruttando diversi moduli python specifici nell’apertura ed elaborazione di file pdf e power-point quali [DA INSERIRE]

In questo modulo potrebbero essere inserite ulteriori sotto-operazioni come la traduzione dei file.

## Indicizzazione

Questa primitiva offre all’utente la possibilità di indicizzare i documenti caricati o modificati […]

Il sistema scansiona il (dbA) ed individua i documenti aventi valore Indexed a false restituendo i loro path.

I path sono utilizzati per accedere al (dbB) ed estrarre il testo delle pagine dei documenti da indicizzare.

Il testo viene utilizzato per aggiornare i serach engines utilizzati dal sistema ovvero Luceene, Topic-Modeling, come approfondito nei paragrafi successivi.

{DA INSERIRE IMG]

## Crawler

Il desktop-crawler implementato ha il compito di navigare il file-system dedicato ed aggiornare i db A (topologia) e gestire il corretto caricamento dei dati nel db B (indice), mediante l’ausilio di moduli dedicati all’estrazione del testo dai diversi tipi di file ed alla pulizia del testo estratto.

[INSERISCI IMMAGINE]

### Aggiornemtno db A

Per l’aggiornamento del db A (topologia) il sistema naviga il file-system e verifica quali documenti necessitano di essere indicizzati. Questi ultimi vengono suddivisi in diverse code (una per ogni tipologia di file supportato) a seconda dell’estensione del file. Le code verranno utilizzate per l’operazione di aggiornamento del db B (indice).

Aggiornamento db B

Sfruttando le code generate nello step precedente il crawler acquisisce i path dei documenti da processare. A seconda della coda (e quindi della tipologia di file) il crawler richiama un modulo specializzato nell’analisi e segmentazione (suddivisione) del file in pagine ed estrazione del testo. Nello specifico, il sistema supporta lettura ed indicizzazione di file di tipo .pptx (.ppt) e .pdf).

Cleaning testo

Una volta generate le pagine del file, queste subiscono un operazione di pulizia del testo. La pipeline di pulizia si compone dei seguenti step:

1. Ricerca
2. Eliminazione stop-words (self-improving set)

# Conclusioni e sviluppi futuri

Abbiamo presentato in questa relazione Lill-AI un sistema user-friendly specializzato nel retrive di documenti di interesse generico per studenti universitari. Lill-AI capitalizza due motori di ricerca distinti i cui risultati vengono combinati per presentare all’utente la lista di pagine semanticamente rilevante alla query sottoposta.

Sicuramente il sistema può essere migliorato incrementando il numero di features considerate in fase di retrieve, ad esempio capitalizzando sulle funzioni NLP di POS o NER. Presentiamo adesso in breve una lista di possibili migliorie da inserire all’interno del sistema per incrementare la sua efficienza.  
  
Tag

L'introduzione dei tag utente sui file memorizzati rappresenterebbe un significativo miglioramento per il nostro motore di ricerca. Questa innovazione consentirebbe agli utenti di arricchire i metadati associati ai propri file, aumentando la precisione delle ricerche e migliorando la pertinenza dei risultati.

Inoltre, l'utilizzo dei tag utente faciliterebbe la categorizzazione e l'organizzazione dei dati personali, promuovendo un'esperienza utente più efficiente e personalizzata.

## NER e POS

L'integrazione di Named Entity Recognition (NER) e Part-of-Speech (POS) tagging nel nostro motore di ricerca apporterebbe notevoli vantaggi. L'impiego del NER consentirebbe l'identificazione e la classificazione di entità come nomi di persone, luoghi e organizzazioni nei documenti archiviati, migliorando così la precisione delle ricerche e facilitando la ricerca di informazioni specifiche in modo più rapido ed efficace.

Allo stesso tempo, l'utilizzo del POS tagging consentirebbe un'analisi più dettagliata della struttura grammaticale dei testi, identificando le parti del discorso come verbi, sostantivi e aggettivi. Questo rafforzerebbe la comprensione del contesto e la capacità del motore di ricerca di restituire risultati più coerenti e pertinenti.

In conclusione, l'adozione di NER e POS tagging costituirebbe un importante passo avanti nell'ottimizzazione del nostro motore di ricerca, migliorando la qualità delle ricerche, l'accuratezza dei risultati e l'esperienza complessiva dell'utente. Ciò contribuirebbe a consolidare la nostra posizione sul mercato e a soddisfare meglio le esigenze degli utenti.

Traduzione automatica