# Documentazione tecnica JNdiff & JNmerge

## Premessa

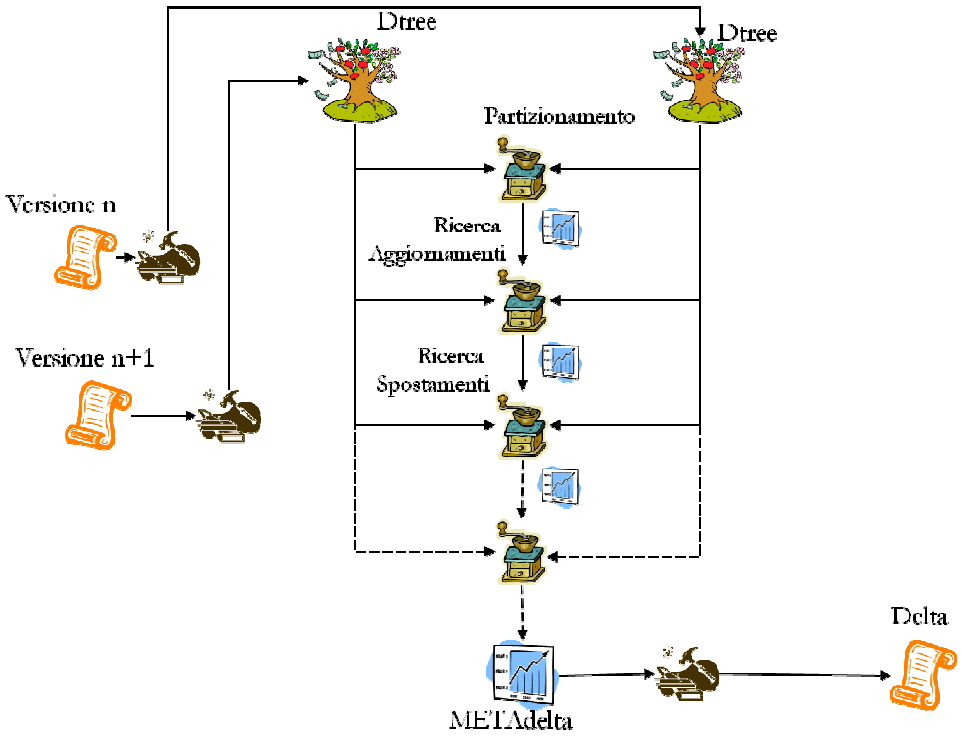
JNdiff e JNmerge sono due applicativi scritti in java che permettono di gestire il confronto di documenti XML, indipendentemente dal DTD dei documenti confrontati. Questi forniscono un nuovo approccio ad diff di documenti xml, permettendo di specificare attraverso dei parametri ad alto livello il tipo di risultato che si vuole ottenere.

In particolare JNdiff di occupa di prendere due documenti XML, dei parametri ad alto livello che specificano la soluzione voluta e costruisce un documento *delta* che elenca le modifiche rilevate. Il documento *delta* è un' elenco di operazione che devono essere apportare alla versione n per ottenere la versione n+1.

JNmerge si occupa di applicare le operazioni di modifica rilevate(delta) al documento originale per ottenere la versione n+1 con del markup che evidenzia le operazione di ricostruzione effettuate.

## Schema di JNdiff

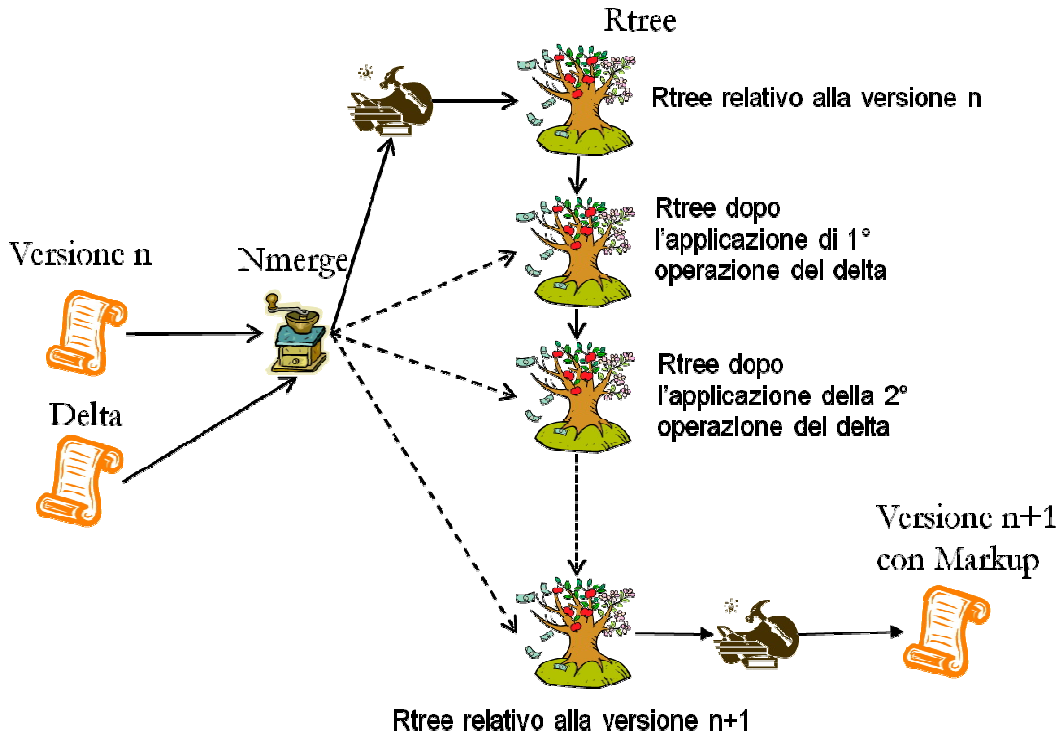
L'algoritmo si compone di una struttura dati capace di mantenere le informazioni relative alle relazioni sui documenti confrontati  e di una serie di fasi  che in maniera sequenziale arricchiscono la relazione tra i nodi dei documenti. Il numero di fasi è arbitrario, ogni fase esamina i due documenti da confrontare ed esprime nuove relazioni sui nodi che li compongono. Alla fine del confronto viene derivato un delta che esprime i cambiamenti avvenuti tra i due documenti XML in input.



Il confronto dei documenti si basa su due strutture dati, chiamate Dtree, che sono estensione della classica visione DOM sui documenti XML. Le strutture Dtree forniscono delle proprietà sugli alberi che ne facilitano le operazioni di confronto.

## Schema di JNmerge

JNmerge parte da una rappresentazione del documento, chiamata Rtree(estensione del DOM), che permette di facilitare le operazioni di inserimento e cancellazioni di sottoalberi, di singoli nodi e modifica del contenuto di nodi di testo con inserimento di markup.




All'inizio JNmerge crea la struttura Rtree sul documento in input. A questo punto applica in cascata le operazioni del delta fino ad ottenere l'Rtree relativo alla versione n+1. A questo punto l'Rtree viene trasformato nel documento XML relativo alla versione n+1 con il markup delle operazioni effettuate.

## Struttura del codice in JAVA

Nell'implementazione in Java sono stati realizzati i seguenti package.

1. **Exception:** Contiene le classi relative alle accezioni che il Diff può sollevare nella fase di confronto
2. **Ndiff:** Contiene tutte le classi relative all'operazione di confronto dei documenti:
    a. **Metadelta:** Contiene le classi per memorizzare il delta in un formato astratto, indipendente dalla codifica xml che si vuole ottenere;
    b. **Phases:** Contiene l'implementazione delle varie fasi di cui si compone l'algoritmo;
    c. **Relation:** Contiene l'implementazione delle strutture dati necessarie a mantenere le relazioni trovate dalle fasi tra i due documenti in input;
3. **Nmerge:** Contiene l'implementazione di JNmerge;

4. **Vdom:** Contiene un'estensione del DOM (vectorDOM) che permette di facilitare i confronti e ricostruire semplicemente i documenti:
    a. **Diffing:** Contiene l'implementazione delle classi per costruire gli alberi durante la fase di confronto;
    b. **Recostruction:** Contiene l'implementazione delle classi per costruire gli alberi che permettono una facile ricostruzione dei documenti;
5. **Debug:** Classi di supporto per il debug dell'applicativo;

## Classi principali

Le classi principali sono due e permettono di confrontare e ricostruire un documento.

1. **ndiff.Ndiff**: E' la classe che effettua il diff. Il metodo di accesso è ***diff(docA, docB)*** che ritorna un domDocument che rappresenta il documento *delta*. Si può utilizzare il diff anche attraverso il metodo ***diff(docA, docB, Delta),*** che salva il delta in formato xml nel file specificato dal parametro Delta. DocA e DocB sono i percorsi dei file xml da confrontare. L'operazione di confronto può sollevare tre tipi di eccezione:
    a. ***InputFileException***: Nel caso in cui ci sono problemi nell'apertura o nel parsing dei file in input. Viene riportato il nome dei file su cui si è verificato il problema;
    b. ***OutputFileException***: Nel caso in cui ci sono problemi nel salvataggio del file contente il risultato. Viene riportato il tipo di problema;
    c. ***ComputePhaseException***: Nel caso si abbia un'eccezione durante una delle fasi di confronto. Viene riportata la fase in cui si è avuta l'eccezione e il tipo di eccezione;

2. **nmerge.Nmerge:** E' la classe che fornisce la funzionalità di ricostruzione del documento. Il metodo di accesso principale è ***merge (DocA, Delta, output),*** dove il *Delta* può essere passato sia in formato DOMdocument che attraverso il percorso del file xml contente il delta.
    a. ***InputFileException***: Nel caso in cui ci sono problemi nell'apertura o nel parsing dei file in input. Viene riportato il nome dei file su cui si è verificato il problema;
    b. ***OutputFileException***: Nel caso in cui ci sono problemi nel salvataggio del file contente il risultato. Viene riportato il tipo di problema;

## Markup inserito da JNmerge

Il markup relativo alle modifiche rilevate viene inserito attraverso degli attributi facenti parte del namespace **http://schirinz.web.cs.unibo.it/Ndiff**, con prefisso **ndiff.** Nel caso in cui il markup si riferisce a parti di documento in cui non è possibile utilizzare attributi(Es. parti di testo), viene usato un elemento **w**(wrapper) che racchiude la parte da marcare.

### *Operazioni con uso del wrapper*

- Inserimento di testo
  … **<ndiff:w status="inserted">** testo da inserire **</ndiff:w>** …
  Il nuovo testo inserito viene racchiuso in un wrapper e viene specificato l'attributo status uguale a "deleted";

- Cancellazione di testo
  … **<ndiff:w status="deleted">** testo rimosso **</ndiff:w> …**
  Il testo che viene segnato come rimosso, viene inserito in un wrapper e viene specificato lo status=”deleted”;

- Spostamento: Punto da cui è stato spostato il testo
  …<**ndiff:w status="movedTo" idref="XX">** …. **</ndiff:w>** …
  Il testo spostato viene racchiuso in un wrapper e viene specificato l’attributo status uguale a “movedTo” e l’attributo idref per collegarlo alla nuova posizione;

- Spostamento: Punto in cui è stato inserito il testo spostato:
  **..<ndiff:w status="movedFrom" id="XX"**> .... </nodo> ..
  Il testo spostato viene racchiuso in un wrapper e viene specificato l’attributo status uguale a “movedFrom” e l’attributo id che mantiene il collegamento relativo allo spostamento.

### *Operazioni senza wrapper*

- Inserimento di sottoalbero:
  <nodo **ndiff:status="inserted"**>...</nodo>
  La segnalazione dell’inserimento di un sottoalbero avviene attraverso l’inserimento dell’attributo *status=“inserted”* nella radice del sottoalbero inserito;

- Cancellazione di sottoalbero:
  <ns:nodo **ndiff:status="deleted"**>...</ns:nodo>
  La cancellazione di un’intero sottoalbero viene segnalata attraverso l’inserimento dell’attributo *ndiff:status=”deleted”* nell’elemento radice del sottoalbero rimosso;

- Spostamento: Punto da cui è stato spostato il sottoalbero:
  <nm:nodo **ndiff:status="movedTo" ndiff:idref="XX"**>....</nm:nodo>
  Il punto da cui è stato spostato il sottoalbero viene segnalato attraverso l’attributo *ndiff:status=”movedTo”,* e viene inserito l’attributo *idref* per mantenere il collegamento con la nuova posizione. Il punto da cui è stato spostato il sottoalbero, rappresenta la posizione di questo nel documento originale.

- Spostamento: Punto in cui è stato spostato il sottoalbero:
  <nodo **ndiff:status="movedFrom" ndiff:id="XX"**> .... </nodo>
  Il punto in cui viene spostato il sottoalbero viene segnalato attraverso l’attributo *ndiff:status=”movedFrom”* e viene inserito l’attributo *id* per mantenere il collegamento con la vecchia posizione. La nuova posizione del sottoalbero rappresenta la posizione di questo nel documento modificato.

### *Modifiche al sottoalbero*

Nei nodi in cui è avvenuta una modifica al sottoalbero, viene inserito l’attributo **ndiff:status=”modified”**

### *Altre operazioni*

Le operazioni di Upgraded(Cancellazione di un nodo),Downgraded(Inserimento di un singolo nodo),Cancellazione di un attributo,Inserimento di un attributo,Cambio del valore di un attributo,

vengono rilevate da JNdiff e riportate nel documento *delta*. Queste vengono applicate da JNmerge, ma attualmente non viene inserito del markup per evidenziare la modifica.

## Esempio di file di configurazione per JNDIFF

Il file di configurazione di jndiff permette di specificare le fasi ed i relativi parametri che l'algoritmo deve eseguire durante la fase di confronto. Per disabilitare una fase basta commentare l'elemento che si riferisce alla fase stessa.

```
<Nconfig>
        <normalize ltrim="true" rtrim="true" collapse="true" emptynode="false" commentnode="false"/>
        <phases>
                <Partition/>
                <FindUpdate level="10"/>
                <FindMove range="10" minweight="31"/>
                <Propagation attsimilarity="40" forcematch="false"/>
        </phases>
</Nconfig>
```

- Normalize-Ltrim: Settiamo a true se vogliamo che nei documenti in input non vengono considerati i "whitespace" che sono all'inizio dei nodi di testo;
- Normalize-Rtrim: Settiamo a true se vogliamo che nei documenti in input non vengono considerati i "whitespace" che sono alla fine dei nodi di testo;
- Normalize-Collapse: Settiamo a true se vogliamo collassare sequenze di "whitespace" in un'unico "whitespace" all'interno dei nodi di testo;
- Normalize-EmptyNode: Settiamo a true se vogliamo considerare nodi di testo contenenti sono "whitespace";
- Normalize-CommentNode: Settiamo a true se vogliamo considerare i nodi commento;
- FindUpdate-Level: Impostiamo la soglia entro la quale due nodi di testo vengono considerati come aggiornati. Il valore è in percentuale e si basa su una funzione di somiglianza tra nodi testuali.
  Es. impostiamo a 100 se vogliamo considerare come aggiornati solo i nodi che sono uguali.
- FindMove-range: Imposta il range di ricerca di eventuali spostamenti. Il range è un'intorno di un nodo, calcolato secondo una funzione.
  Es. Impostiamo a 100 se vogliamo cercare eventuali spostamenti in tutto il documento.
- FindMove-MinWeight: Imposta il "peso" che deve avere un nodo per essere considerato spostato. Il peso di un nodo è determinato dal numero di caratteri presenti nel suo sottoalbero.
  Es. Impostiamo a 30 se vogliamo non considerare come spostati, parti di documento con testo inferiore a 30 caratteri.
- Propagation-AttSimilarity: Imposta il livello di somiglianza che devono evere due **singoli** nodi per poter essere considerati uguali.
  La somiglianza di due singoli nodi si basa sugli attributi e sul contenuto dei frammenti riconosciuti nel sottoalbero.
- Propagation-forcematch: Se abilitata, durante il confronto viene dato più peso alla struttura del documento che al suo contenuto.

## Uso di JNdiff&JNmerge da riga di comando

Jndiff può essere utilizzato attraverso riga di comando in tre modalità:

1. Calcolo del diff tra due documenti XML, specificando il file di configurazione da utilizzare. Se il file di configurazione non viene specificato vengono usati dei parametri di default.
   java -jar JNdiff.jar diff [-c config.xml] <Original> <Modified> <Ndelta>
   - Config.xml : Percorso del file di configurazione per JNdiff
   - Original : Percorso del documento xml originale(Versione n)
   - Modified : Percorso del documento xml modificato(Versione n+1)
2. Applicazione del delta alla *versione n* per ottenere la *versione n+1* con il markup relativo alle operazione di modifica rilevate. Eventualmente si può processare il risultato del diff attraverso un foglio di stile XSLT.
   java -jar JNdiff.jar merge <Original> <Ndelta> <Nmarkup> [-xslt <xslt> <output>]
   - Original : Percorso del documento xml originale(Versione n)
   - Modified : Percorso del documento xml modificato(Versione n+1)
   - Nmarkup : Percorso del documento di output.
   - Xslt : Percorso del file xslt che si vuole applicare all'output
   - Output : Percorso del documento di output relativo alla trasformazione XSLT del documento Nmarkup.
3. Calcolo del Diff e applicazione diretta delle modifiche rilevate.
   java -jar JNdiff.jar diff&merge [-c config.xml] <Original> <Modified> <Nmarkup> [-xslt <xslt> <output>]
   - Config.xml : Percorso del file di configurazione per JNdiff
   - Original : Percorso del documento xml originale(Versione n)
   - Modified : Percorso del documento xml modificato(Versione n+1)
   - Nmarkup : Percorso del documento di output.
   - Xslt : Percorso del file xslt che si vuole applicare all'output
   - Output : Percorso del documento di output relativo alla trasformazione XSLT del documento Nmarkup.