

In collaborazione con



[www.futuresoftware.it](http://www.futuresoftware.it)

[info@futuresoftware.it](mailto:info@futuresoftware.it)

# Progetto Conversione XML2PDF: PowerConversion Documentazione

Progetto: Conversione XML2PDF

Autori: Ing. Marco Maltraversi

Versione:

1.0

Stato:

Completa

## Indice

---

Indice.....	2
1. Introduzione .....	3
2. Descrizione del processo di trasformazione da XML a PDF .....	4
3. Come inserire FOP in una applicazione JAVA.....	6
4. Manuale applicazione software.....	8
4.1 Descrizione dell'applicazione realizzata .....	8
4.2 Le Classi.....	12
4.3 Requisiti .....	13
4.4 Come si installa ed esegue .....	13
4.5 Stato dell'arte per la trasformazione xml2pdf.....	14
5. Risorse utili .....	14

## 1. Introduzione

---

Questo documento contiene tutta la documentazione prodotta durante lo svolgimento del progetto Conversione XML2PDF - PowerConversion. L'obiettivo del progetto è la realizzazione di un'applicazione per la trasformazione di documenti XML in documenti PDF, tramite l'utilizzo di XSLT.

XML rappresenta un "metalinguaggio", ovvero un insieme di regole e convenzioni che consentono di descrivere qualunque linguaggio di markup, basato quindi su marcatori, per le esigenze del momento, siano esse descrittive e progettuali oppure prettamente operative.

Questa idea deriva in parte dai concetti di base di SGML (*Standar Generalized Language*), che aveva esattamente questi obiettivi. Il vantaggio di XML su SGML è dato dalla semplicità con la quale in XML è possibile definire una struttura gerarchica per le informazioni, semplicità che si traduce in velocità di apprendimento da parte dell'utente e quindi in maggiore possibilità di diffusione.

Con XML è possibile scrivere il proprio linguaggio definendo, anche se ciò non è obbligatorio, le relazioni di gerarchia tra le varie componenti (tramite una grammatica DTD, ovvero *Document Type Definition*), fino ad arrivare alla determinazione di vere e proprie grammatiche che consentono di verificare la correttezza semantica dei documenti XML.

Un'altra caratteristica di questo linguaggio, è inoltre la capacità che ha di essere trasformato in qualunque altra cosa: una differente applicazione XML, una pagina web (X)HTML, un documento PDF o PostScript e, in genere, ogni formato per il quale esista un filtro di trasformazione.

Ecco perchè XML viene utilizzato sia facendo riferimento ai tanti linguaggi che sono stati messi a disposizione dagli organismi che definiscono gli standard internazionali (come ad esempio il W3C, che ha scritto le specifiche XHTML), sia come linguaggio di interscambio di informazioni tra applicazioni diverse per le quali sono state definite strutture grammaticali proprietarie.

Scrivere in XML significa da un lato garantirsi la possibilità di creare ed estendere il proprio linguaggio, e dall'altro avere a disposizione il supporto di strumenti sempre più evoluti per l'editing, la validazione della grammatica e la trasformazione dei documenti in formati diversi.

---

## 2. Descrizione del processo di trasformazione da XML a PDF

---

Il World Wide Web Consortium ha stabilito una specifica per XSL, che è stata divisa in due parti:

- XSLT, un linguaggio per trasformare documenti XML;
- XSL Formattino Objects (XSL FO), un vocabolario<sup>1</sup> XML per specificare la semantica di formattazione;

XSL è l'acronimo di *eXtensible Style Language*. Si tratta della componente di un linguaggio per la definizione di fogli di stile che consentono di gestire la presentazione di dati contenuti in documenti XML, così che questi possano essere visualizzati correttamente, per esempio in un browser web.

XSLT identifica la componente di XSL che si occupa direttamente di trasformare un documento XML in un altro tipo di documento e costituisce quindi il primo strumento utilizzato per poter gestire la presentazione dei dati XML.

Il processore XSLT è un software che prende un documento XSLT e lo applica ad un documento XML producendo in output un documento di tipo diverso, che può essere nuovamente un documento XML, del semplice testo oppure un documento testuale dalla stesura complessa.

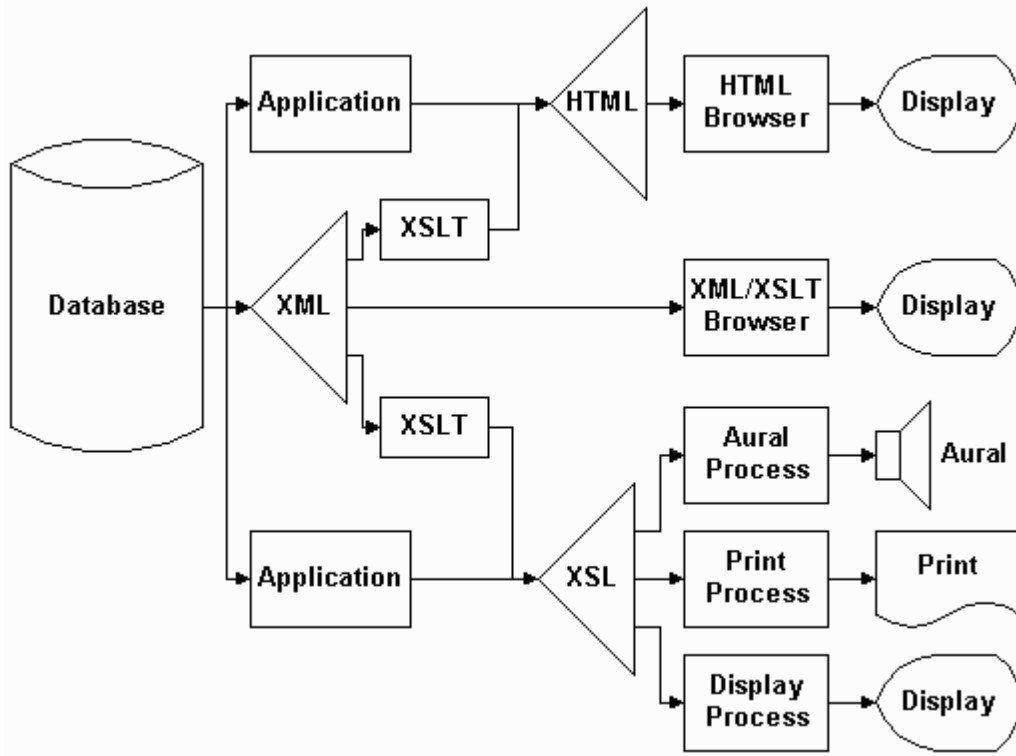
Una delle trasformazioni che si incontrano più di frequente è quella da XML a HTML; nel caso specifico, un documento XML generato da un'applicazione viene trasformato in un documento HTML che possa essere visualizzato correttamente da un browser web<sup>2</sup>.

---

<sup>1</sup> In XML è possibile definire uno spazio dei nomi personale: Un meccanismo che consente agli sviluppatori di qualificare in modo univoco i nomi degli elementi e le relazioni e di rendere questi nomi riconoscibili. In questo modo è possibile evitare conflitti di nomi su elementi che presentano lo stesso nome ma sono definiti in vocabolari diversi. I vocabolari consentono di combinare tag appartenenti a più spazi dei nomi. Ciò è essenziale se i dati provengono da origini diverse. Gli spazi dei nomi assicurano che i nomi degli elementi non siano in conflitto e chiariscono a quale termine si riferiscono.

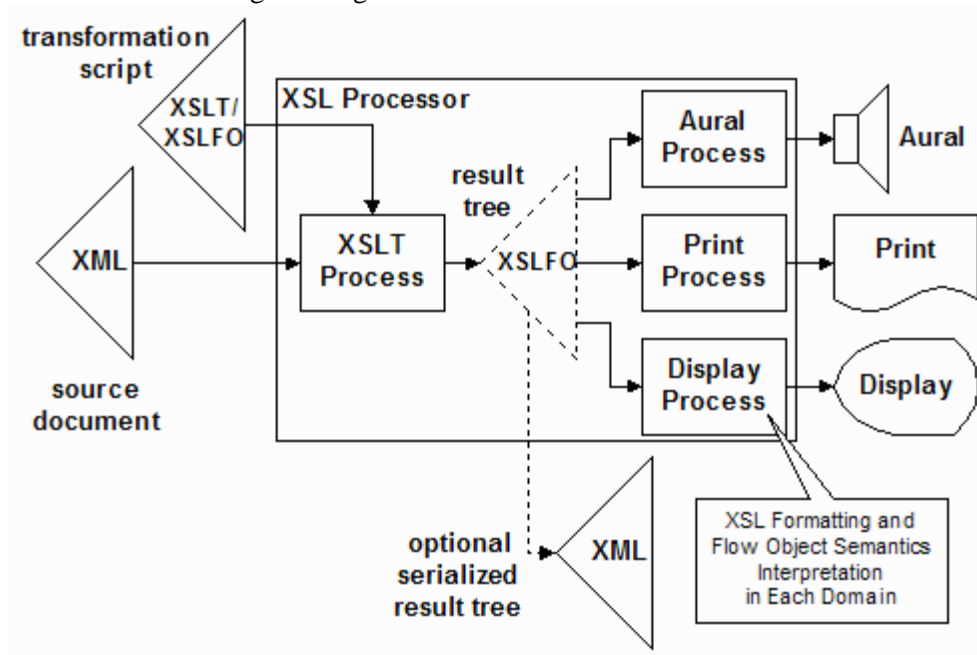
Uno spazio dei nomi identifica un vocabolario XML definito all'interno di un URN. Un attributo su un elemento, attributo o riferimento dell'entità associa un nome breve all'URN che definisce lo spazio dei nomi. Questo nome abbreviato viene quindi utilizzato come prefisso per il nome dell'elemento, dell'attributo o del riferimento dell'entità per identificare in modo univoco lo spazio dei nomi. I riferimenti agli spazi dei nomi hanno un ambito. Tutti i nodi figlio al di sotto del nodo che specifica lo spazio dei nomi ereditano quello spazio dei nomi. Ciò consente ai nomi non qualificati di utilizzare lo spazio dei nomi predefinito. Vedere anche Spazio dei nomi RDF.

<sup>2</sup> Un esempio della crescente importanza che XML sta riscuotendo nel mondo del 3W può essere dato dal fatto che anche la gestione di siti con contenuti dinamici (blog, forum, giornali online..) viene fatta utilizzando XML; infatti, grazie alla assoluta portabilità di questo linguaggio, è possibile separare la gestione dei contenuti, dalla gestione della grafica del sito.

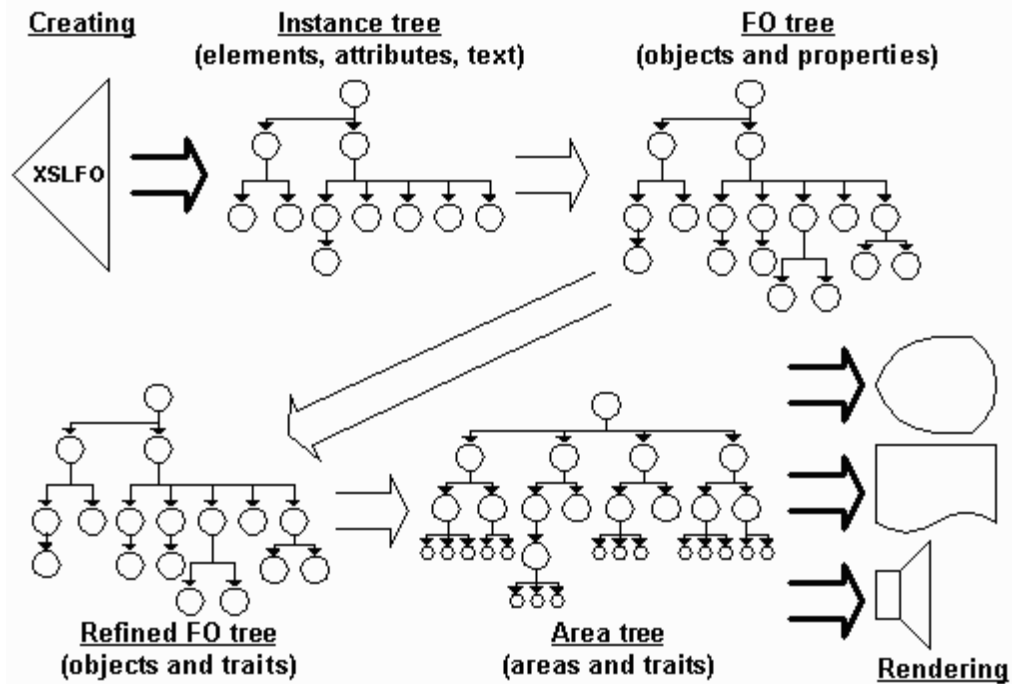


XSL Formatting Object è esso stesso un linguaggio basato su XML, che permette all'utente di specificare, con grande dettaglio, la paginazione, il layout e lo stile che possiamo applicare ad un contenuto. Questo linguaggio risulta essere un po' più complesso del precedente, in quanto molto prolisso;

Il processo di trasformazione da XML a PDF (ma in generale, anche qualsiasi altro tipo di trasformazione) può essere schematizzato dalla seguente figura:



Il processo di modellazione di un file XSLFO è invece illustrato nella seguente figura:



### 3. Come inserire FOP in una applicazione JAVA

Per poter utilizzare FOP<sup>3</sup> in una applicazione personale, è necessario ottenere una istanza della classe `org.apache.fop.apps.Driver`. Questo oggetto potrà essere impiegato per lanciare più operazioni di rendering. Per ogni operazione di trasformazione, è necessario creare un'istanza di `org.apache.fop.apps.Fop` tramite uno dei metodi della classe `FopFactory`. Nella chiamata al metodo, sarà possibile specificare il formato dell'output da utilizzare e se il tipo di rendering selezionato richiede un `OutputStream`, quale `OutputStream` per il risultato del rendering. Durante uno dei possibili processi di trasformazione, è possibile indicare a FOP, una configurazione personale tramite una istanza di `FoUserAgent`.

Apache FOP si basa su JAXP (la **Java API for XML Processing** che permette alle applicazioni di eseguire il parsing e la trasformazione di documenti XML, indipendentemente dalla particolare implementazione di XML). Utilizza gli eventi generati dal parser SAX per costruire l'albero associato al documento XSL-FO fornito come input o come risultato della trasformazione di un documento XML tramite un documento XSLT.

Il codice java che permette di ottenere un file pdf da un XSL-FO è il seguente:

```
import org.apache.fop.apps.FopFactory;
import org.apache.fop.apps.Fop;
import org.apache.fop.apps.MimeConstants;

/*...*/

// Step 1: Construct a FopFactory
// (reuse if you plan to render multiple documents!)
FopFactory fopFactory = FopFactory.newInstance();

// Step 2: Set up output stream.
```

<sup>3</sup> Per ulteriori informazioni in merito al progetto FOP, consigliamo di visitare il sito <http://xml.apache.org/fop>; Qui di seguito verrà presentato un breve excursus su come utilizzare i file di esempio forniti dal progetto FOP, per realizzare le trasformazioni oggetto del progetto PowerConversion.

```
// Note: Using BufferedOutputStream for performance reasons (helpful with
// FileOutputStreams).
OutputStream out = new BufferedOutputStream(new FileOutputStream(new
// File("C:/Temp/myfile.pdf")));

try {
    // Step 3: Construct fop with desired output format
    Fop fop = fopFactory.newFop(MimeConstants.MIME_PDF, out);

    // Step 4: Setup JAXP using identity transformer
    TransformerFactory factory = TransformerFactory.newInstance();
    Transformer transformer = factory.newTransformer(); // identity transformer

    // Step 5: Setup input and output for XSLT transformation
    // Setup input stream
    Source src = new StreamSource(new File("C:/Temp/myfile.fo"));

    // Resulting SAX events (the generated FO) must be piped through to FOP
    Result res = new SAXResult(fop.getDefaultHandler());

    // Step 6: Start XSLT transformation and FOP processing
    transformer.transform(src, res);

} finally {
    //Clean-up
    out.close();
}
```

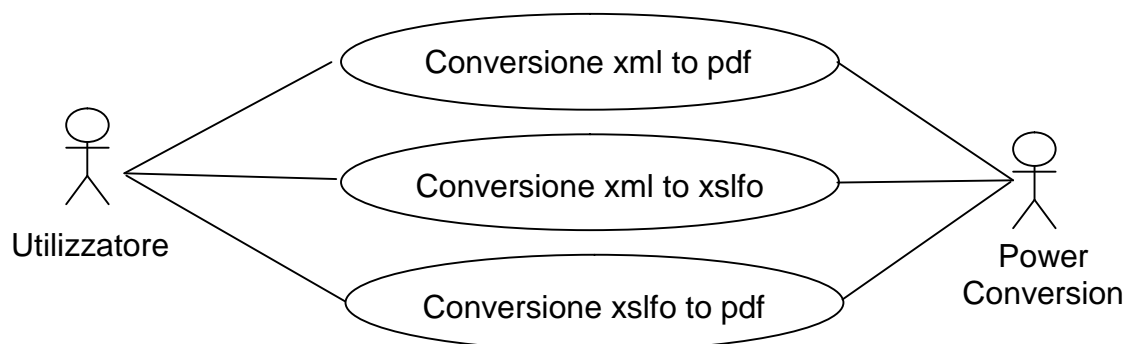
L'applicazione realizzata si basa sugli esempi forniti insieme al programma FOP, alla locazione "{fop-dir}/examples/embedding".

## 4. Manuale applicazione software

Questa sezione si occupa di descrivere in dettaglio l'applicazione realizzata per la manipolazione di documenti XML. In particolare, allo stato attuale PowerConversion è in grado di realizzare le seguenti trasformazioni/conversioni: da xml a pdf, da xml a xslfo, da xslfo a pdf;

### 4.1 Descrizione dell'applicazione realizzata

La figura seguente riporta i principali casi d'uso:



#### USECASE 1:

In questo caso, oltre al file XML che si vuole convertire, sarà necessario anche indicare un foglio di stile (XSLT), per permettere la generazione di un file XSLFO (file di impaginazione/formattazione), che verrà direttamente elaborato da FOP e che darà origine al file PDF.

#### USECASE 2: (\*)

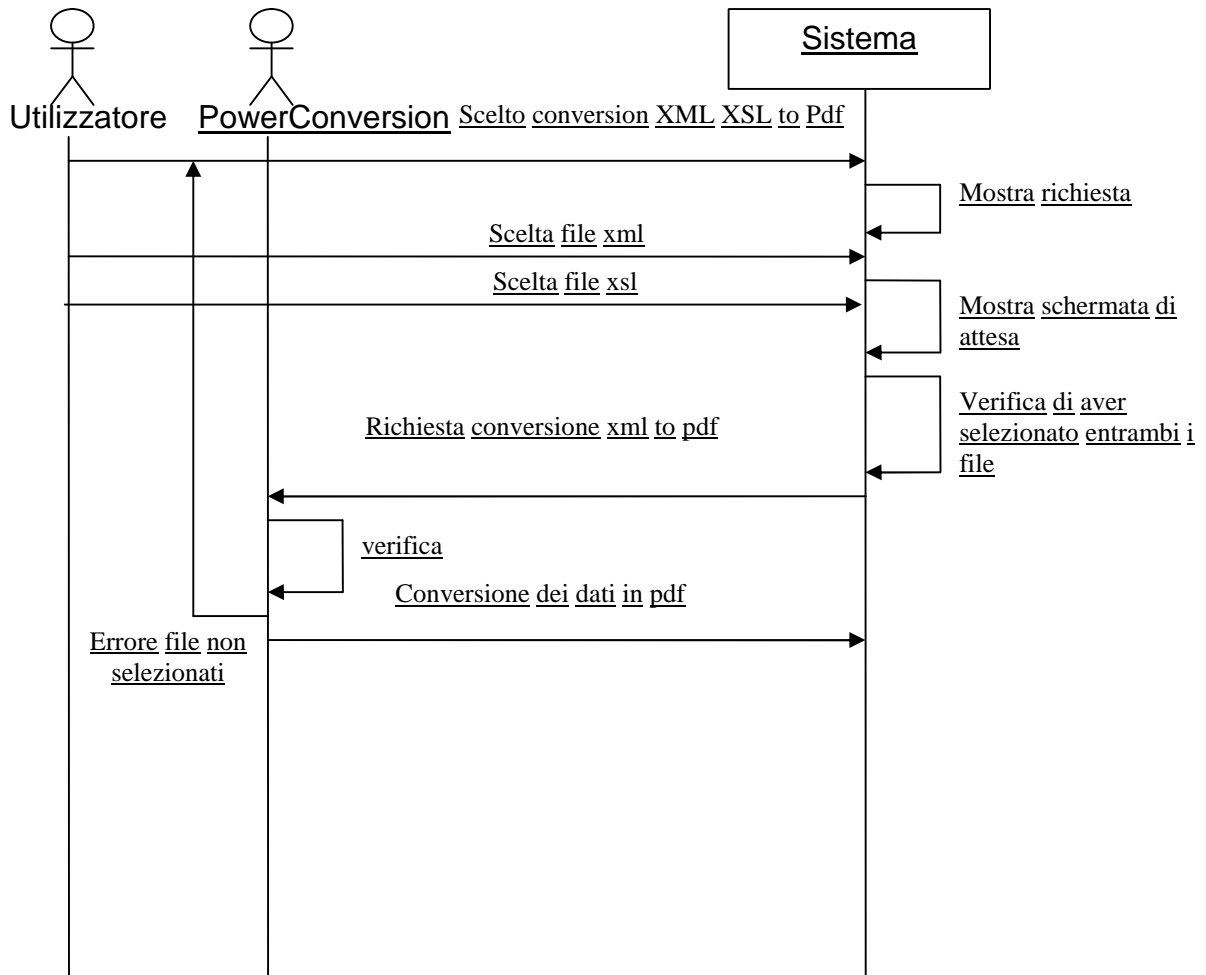
In questo caso, il file XML viene trasformato in un documento di impaginazione di tipo XSLFO;rispetto allo usecase precedente, cambia solo il formato di destinazione;

#### USECASE 3: (\*)

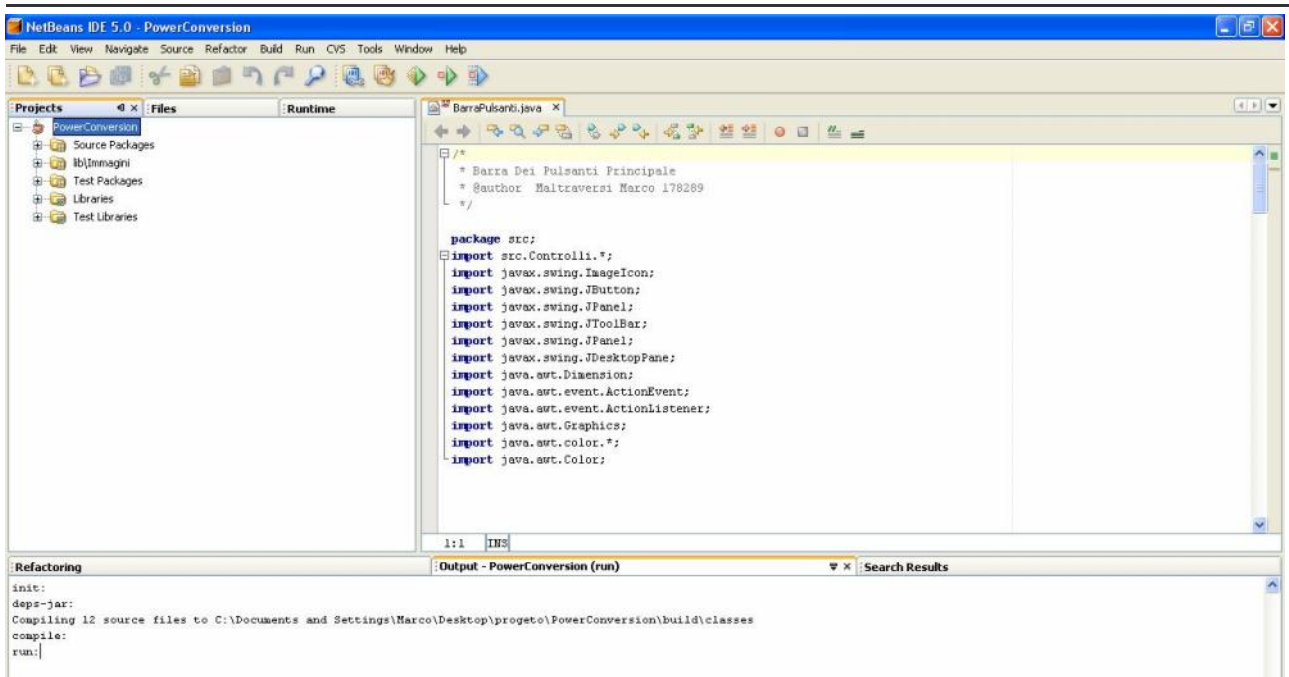
In questo caso, il file di origine è unico e di tipo XSLFO, che verrà in seguito trasformato dal programma FOP, inglobato all'interno di PowerConversion, in un file PDF;

\*Questi due casi d'uso sono il risultato intermedio della trasformazione precedente;

Viene anche riportato il sequenze diagram di un caso d'uso:



Il progetto è stato realizzato utilizzando NetBeans IDE 5.0:

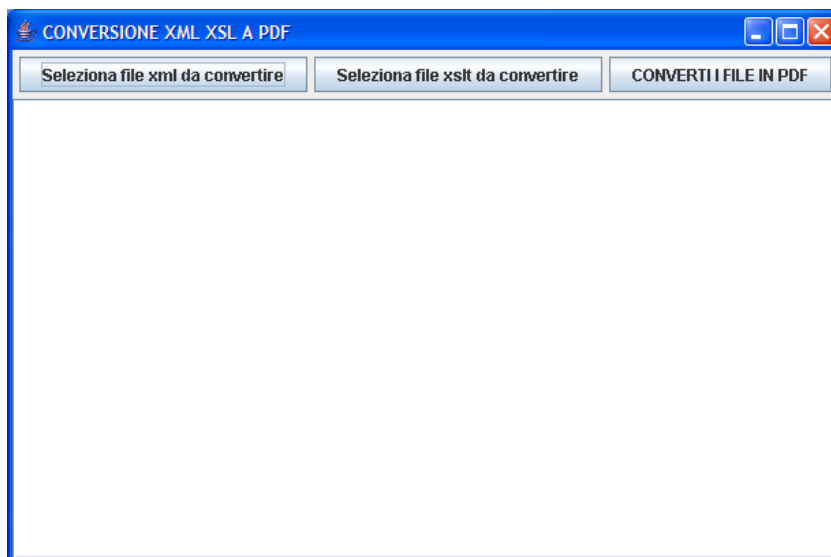


**Figura 0-1: Progetto PowerConversion in NetBeans**

La schermata principale del programma consente all'utente di selezionare l'opzione desiderata.

**Figura 0-2: Schermata Principale**

Ogni operazione eseguita è monitorata e le varie azioni vengono salvate in un file di testo che viene creato nella directory principale del progetto.



**Figura 0-3: Schermata per la conversione XML to PDF**

Per proseguire nella conversione è necessario selezionare il file xml e xsl. E' stata quindi realizzata un'interfaccia grafica che permette di selezionare i file dal PC; sono stati applicati dei filtri per permettere la selezione dei soli file richiesti.

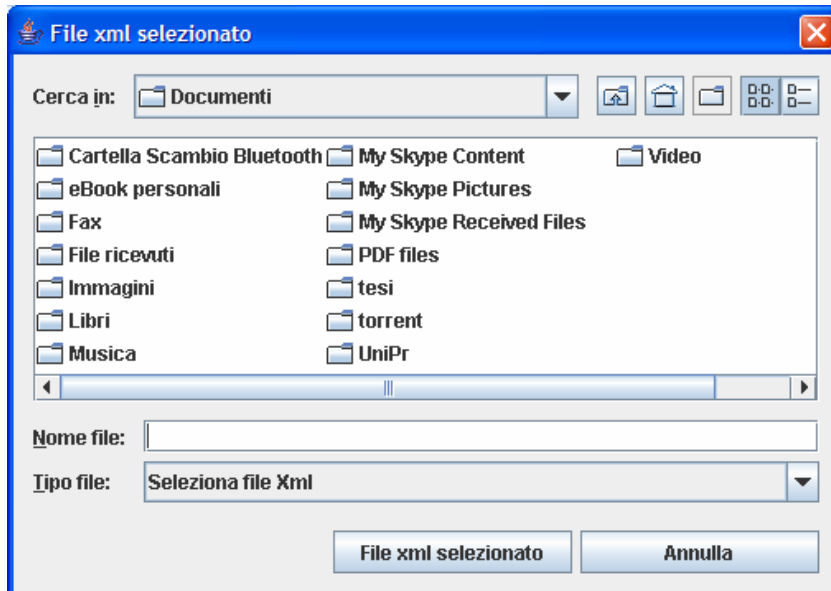


Figura 0-4: Selezione dei file

Una volta selezionati i file è possibile eseguire la conversione, che creerà una directory all'interno di PowerConversion, dello stesso nome del file xml e pdf appena generato.

Vengono poi presentate, anche le schermate delle altre due conversioni che il programma è in grado di eseguire:

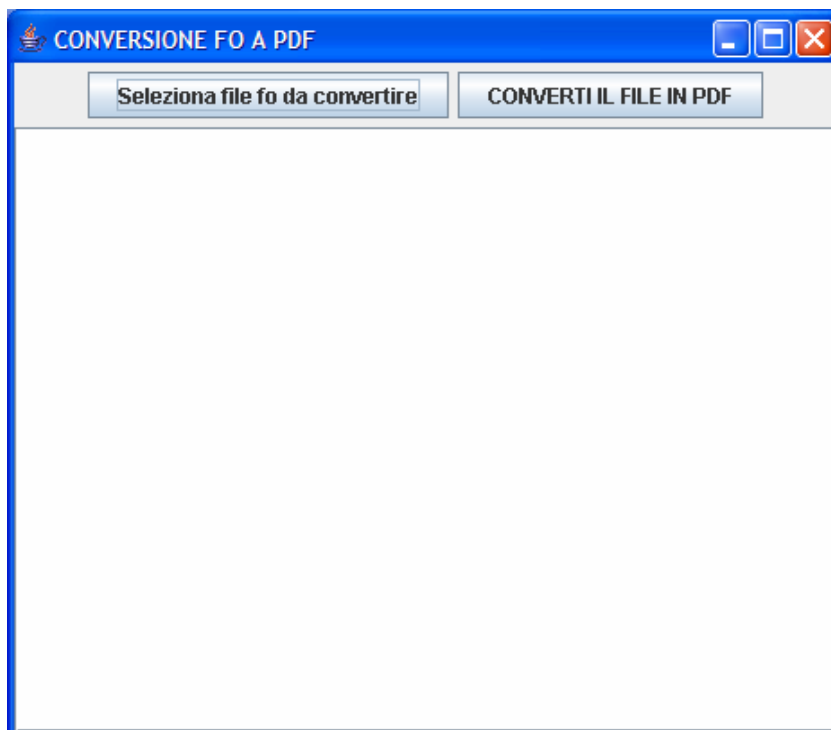
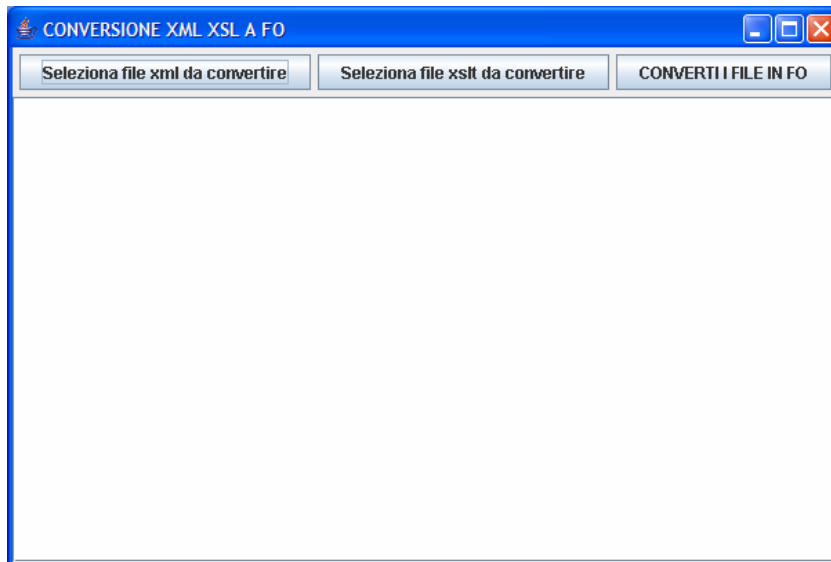


Figura 0-5: Schermata per la conversione da XSLFO a PDF



**Figura 0-6: Schermata per la conversione da XML a XSLFO**

La chiusura dell'applicazione può essere effettuata, ogni qual volta viene presentata la schermata iniziale, tramite l'introduzione di uno '0' oppure dopo l'esecuzione del punto uno; in tutti gli altri casi, l'applicazione continua a fornire il suo servizio.

## **4.2 Le Classi**

Le classi realizzate si trovano nella cartella *src/* dell'applicazione; di seguito verrà data una breve descrizione delle classi principali e del compito associato a ciascuna di queste:

Classe	Descrizione
Main.java	Classe principale del progetto;
BarraPulsanti.java	Classe che si occupa della barra contenuta nella schermata principale;
MainForm.java	Classe che si occupa dell'interfaccia principale del Programma;
GUI.java	Interfaccia che definisce le funzionalita' dell'interfaccia utente
Controller.java	Questa classe si occupa di verificare che vengano selezionati correttamente i file per la conversione da xml a pdf (in questo caso i file hanno estensione xml e xsl); In particolare, se il check è OK, allora verrà invocata la classe Conversione1XMLPDF.java;
Controller1.java	Questa classe si occupa di verificare che vengano selezionati correttamente i file per la conversione da xslfo a pdf (in questo caso i file hanno estensione xslfo); In particolare, se il check è OK, allora verrà invocata la classe Conversione2FOPDF.java;
Controller2.java	Questa classe si occupa di verificare che vengano selezionati correttamente i file per la conversione da xml a pdf (in questo caso i file hanno estensione xml e xsl); In particolare, se il check è OK, allora verrà invocata la classe Conversione3XMLFO.java;
Conversione1XMLPDF.java	Permette di effettuare la vera conversione da XML-XSL a PDF
Conversione2FOPDF.java	Permette di effettuare la vera conversione da FO a PDF
Conversione3XMLFO.java	Permette di effettuare la vera conversione da XML-XSL a FO
Controller1.java	Classe che controlla passo a passo lo stato della conversione e memorizza in un file di testo tutte le fasi principali dell'elaborazione nella conversione da da XML-XSL a PDF
Controller2.java	Classe che controlla passo a passo lo stato della conversione e memorizza in un file di testo tutte le fasi principali dell'elaborazione nella conversione da da FO a PDF
Controller3.java	Classe che controlla passo a passo lo stato della conversione e memorizza in un file di testo tutte le fasi principali dell'elaborazione nella conversione da da XML-XSL a FO
xmlFilter.java	Filtro che permette l'apertura di soli file Xml
xsltFilter.java	Filtro che permette l'apertura di soli file Xsl
foFilter.java	Filtro che permette l'apertura di soli file Fo
Utils.java	Permette di selezionare uno dei filtri nell'esecuzione dell'applicativo

### 4.3 Requisiti

Sistema Operativo	Pacchetti Software
<b>Applicazione: PowerConversion_v1.0</b>	
Tutti i sistemi operativi che supportano Java	J2sdk1.4.2_01 e successive, j2re

### 4.4 Come si installa ed esegue<sup>4</sup>

L'applicazione non prevede alcuna installazione;

Per la sua esecuzione sono è possibile utilizzare il file batch run.but, assicurandosi che il percorso contenuto al suo interno per raggiungere il file PowerConversion.jar sia corretto, oppure da shell, posizionandosi nella directory principale dell'applicativo, lanciando:

<sup>4</sup> Per la compilazione di PowerConversion, è disponibile il file di progetto per NetBeans, che basterà aprire per poter eseguire le opzioni build e run;

```
java -jar dist\PowerConversion.jar
```

## 4.5 Stato dell'arte per la trasformazione xml2pdf

Tra i programmi che abbiamo incontrato durante lo svolgimento di questo progetto citiamo:

- [Antenna House XSL Formatter](#) (Tool per il rendering di file XSL FO)
- [RenderX](#) (a batch XSLFO rendering)
- [XmlSpy](#) di Altova.com
- [XF Rendering Server 2005](#) di Ecrion.com (XML to PDF Engine)
- [XEP4](#)
- [pdf2text.com](#) offre invece la soluzione opposta, per ottenere file testuali o XML da PDF
- [PDFConstructor](#)
- [<oXygen/> XML Editor & XSLT Debugger](#)

## 5. Risorse utili

---

Per la realizzazione della seguente documentazione e del progetto PowerConversion, sono state utilizzate le seguenti risorse:

- Libri: “XML” di Massimo Canducci, Casa Editrice Apogeo; Questo Libro fornisce gli strumenti di base per la conoscenza del mondo XML;
- WWW:
  - <http://xmlgraphics.apache.org/fop/> (Home del progetto Apache FOP)
  - <http://xmlgraphics.apache.org/fop/resources.html> (Sempre dal sito del progetto FOP, da cui è possibile trovare link a articoli, libri e tutorial online)
  - <http://www.mokabyte.it/2005/07/mokacms-6.htm>  
<http://www.mokabyte.it/2005/09/mokacms-7.htm> (Esempio di utilizzo di XSL FO per la realizzazione di una pagina di articolo)
  - <http://www.xml.com/pub/a/2001/01/17/xsl-fo/index.html>  
<http://www.xml.com/pub/a/2001/01/24/xsl-fo/index.html> (Articolo dedicato sempre all'utilizzo di XSL FO, ma che spiega in modo più dettagliato rispetto a quello di Mokacms, tratto dal capitolo 18 del libro “XML bible”  
<http://www.cafeconleche.org/books/bible2/chapters/ch18.html>)  
<http://xml.com/pub/q/transformingxml> (Utile strumento da utilizzare come tutorial per apprendere il linguaggio XSLT)  
<http://www.xml.com/pub/a/2002/03/20/xsl-fo.html> (Documento che tratta nel dettaglio XSL FO.. seguono  
<http://www.xml.com/pub/a/2000/08/holman/index.html>  
<http://www.xml.com/pub/a/98/10/guide0.html>)
  - <http://a2.pluto.it/a21.htm> (Appunti di informatica libera di Daniele Giacomini)
  - <http://www.w3.org/XML/> e <http://www.w3.org/TR/xslt> dal sito ufficiale della W3C
  - <http://www.w3schools.com/xml/> e <http://www.w3schools.com/xsl/> (Tutorials)