InformaticaUmanistica

Lezione 12-13
Funzionalità avanzate di Greenstone

Pasquale Savino

ISTI - CNR

UNIVERSITÀ DI PISA

Sommario

- **Processo di funzionamento di importazione e building di una collazione**
  - Import
  - Build
- **Il configuration file**
- **Uso di**
  1. Plug-in
  2. Classifiers
  3. Indici
- **Formattazione delle pagine web**

Digital Library Collections

- **Vi è una distinzione tra**
  - COSTRUIRE una collezione
  - FORNIRE informazioni agli utenti
- **È la stessa distinzione che esiste tra il 'compile-time' ed il 'runtime' nei linguaggi di programmazione**
- **La fase di costruzione è necessaria per preparare tutte le strutture dati che vengono poi utilizzate nella fase di delivery delle informazioni**

Costruzione manuale delle Collezioni

Costruzione di una collezione

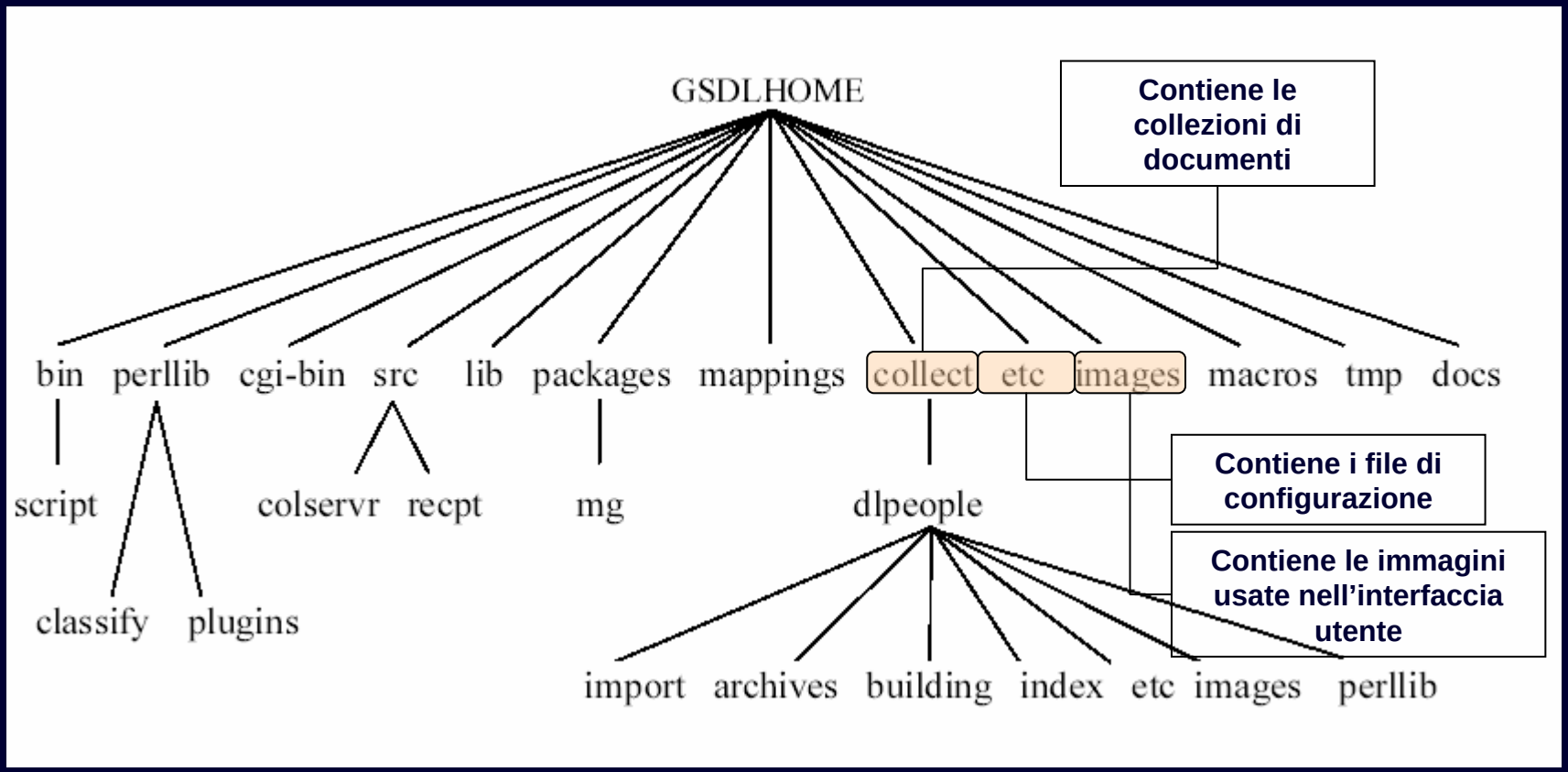
- **Il processo che consiste nel prendere un insieme di documenti ed i metadati che li descrivono e creare tutti gli indici e le strutture dati che ne supportano la ricerca (search), il browsing, e la visualizzazione**

Costruzione di una collezione

- **La costruzione di una collezione prevede quattro fasi**
  - Make
    - ➔ Creare uno scheletro di strutture e di file nel quale verranno inseriti i dati della collezione
  - Import
    - ➔ Importare i documenti ed i metadati e convertirli nel formato Greenstone
  - Build
    - ➔ Costruire gli indici e le strutture dati richieste
  - Install
    - ➔ Rendere operativa la collezione

Make

- Vengono create le seguenti cartelle (directories)

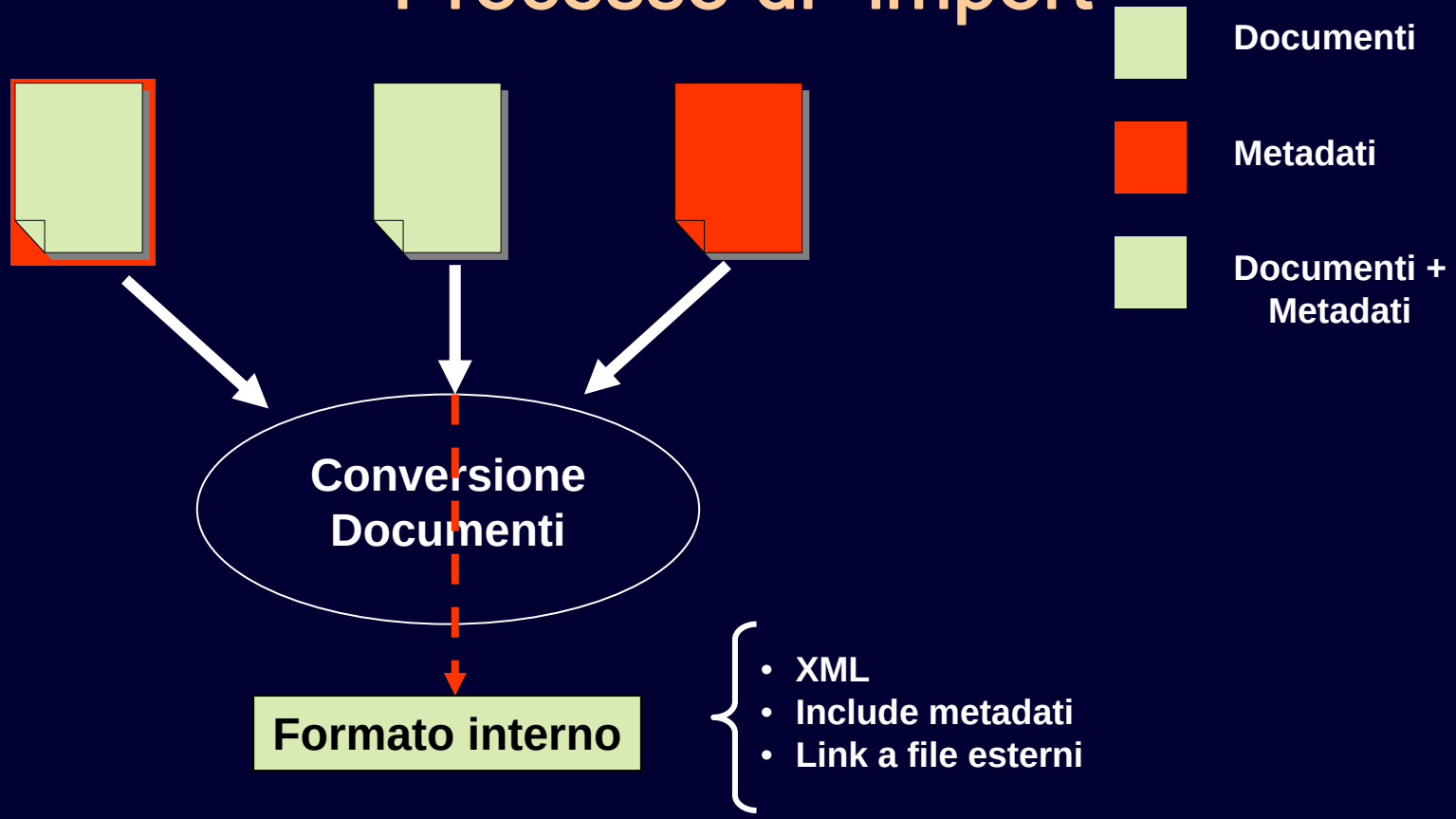


bin	Executable code, including binaries in the directory with your O/S name.
bin/script	Perl scripts used for creating and building collections (for example *import.pl* and *buildcol.pl*). To get a description of any of these programs, type their name at the command prompt.
perllib	Perl modules used at import and build time (plugins, for example).
perllib/plugins	Perl code for document processing plugins.
perllib/classify	Perl code for classifiers (for example the AZList code that makes a document list based on the alphabetical order of some attribute).
cgi-bin	All Greenstone CGI scripts, which are moved to the system cgi-bin directory.
tmp	Directory used by Greenstone for storing temporary files.
etc	Configuration files, initialisation and error logs, user authorisation databases.
src	C++ code used for serving collections via a web server.
src/colservr	C++ code for serving collections—answering queries and the like.
src/recpt	C++ code for getting queries from the user interface and formatting query responses for the interface.
packages	Source code for non-Greenstone software packages that are used by Greenstone.
packages/mg	The source code for MG, the compression and indexing software used by Greenstone.
mappings	Unicode translation tables (for example for the GB Chinese character set).
macros	The macro files used for the user interface.
collect	Collections being served from this copy of Greenstone
lib	C++ source code used by both the collection server and the receptionist.
images	Images used in the user interface.
docs	Documentation.

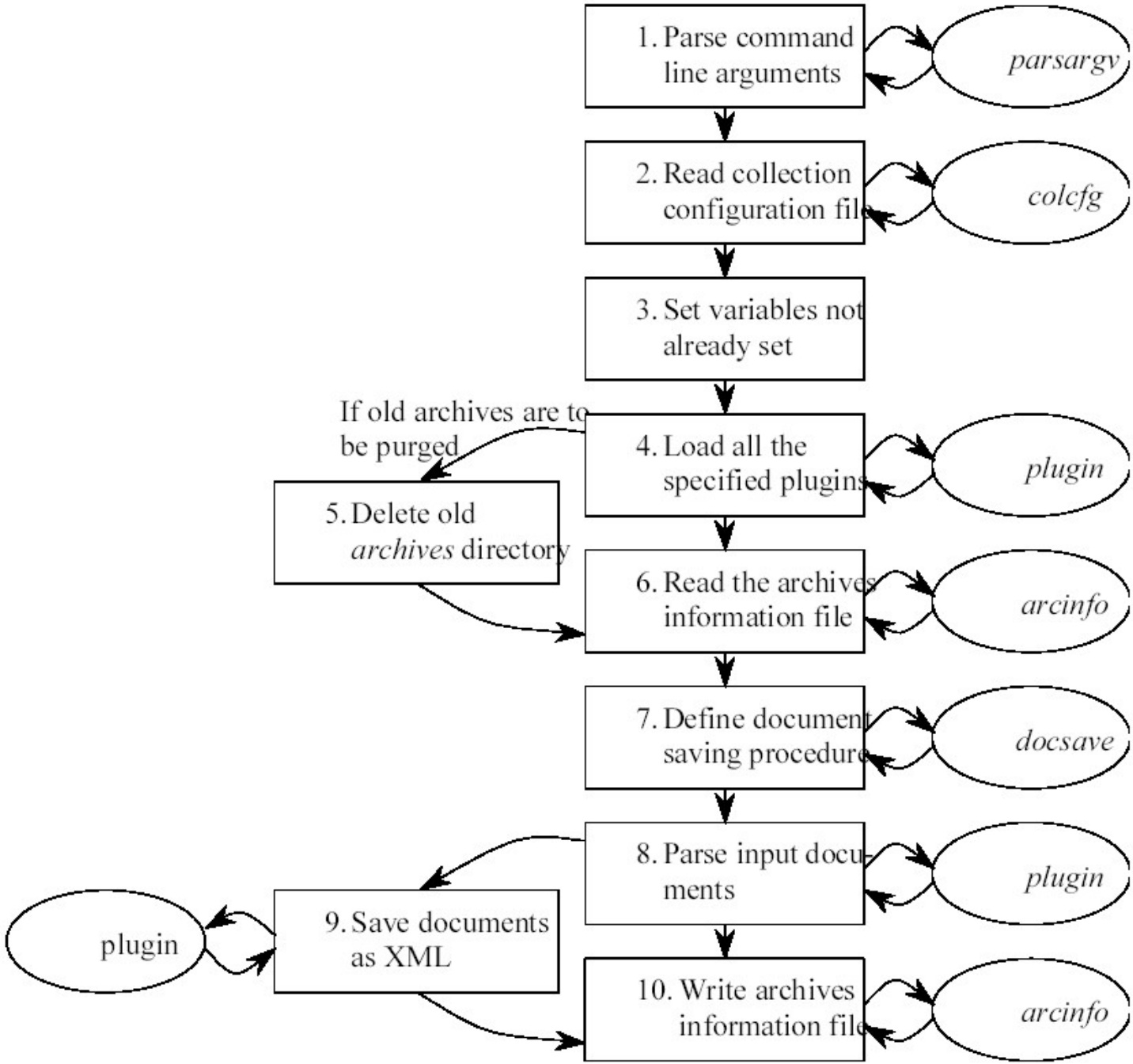
Processo di importazione

- Inserisce i documenti ed i metadati nel sistema in un formato XML standard.
- I documenti originali sono inseriti nella cartella *import*
- Il processo di “import” inserisce i file in formato XML standard nella cartella *archives*
- A questo punto i documenti originali possono essere cancellati
  - Nel caso la collezione debba essere rigenerata, questo può essere fatto a partire dai documenti archiviati
- Ogni nuovo documento da aggiungere alla collezione viene inserito nella cartella *import.* Il processo di importazione viene ripetuto
- Per conservare il formato originale dei documenti, non bisogna cancellare i file in archivio

Processo di “import”



Il processo "import"



Opzioni per “import”

–verbosity	Number 0–3	Control how much information about the process is printed to standard error; 0 gives a little, 3 gives lots.
–archivedir	Directory name	Specify where the Greenstone archive files are stored—that is, where *import.pl* puts them and where *buildcol.pl* finds them. Defaults to *GSDLHOME/collect/col_name/archives*
–maxdocs	Number >0	Indicates the maximum number of documents to be imported or built. Useful when testing a new collection configuration file, or new plugins.
–collectdir	Directory name	Specify where the collection can be found. Defaults to *GSDLHOME/collect*
–out	Filename	Specify a file to which to write all output messages, which defaults to standard error (the screen). Useful when working with debugging statements.
–keepold	None	Do not remove the result of the previous import or build operation. In the case of import, do not remove the contents of the *archives* directory; when building, do not remove the content of the *building* directory.
–debug	None	Print plugin output to standard output.

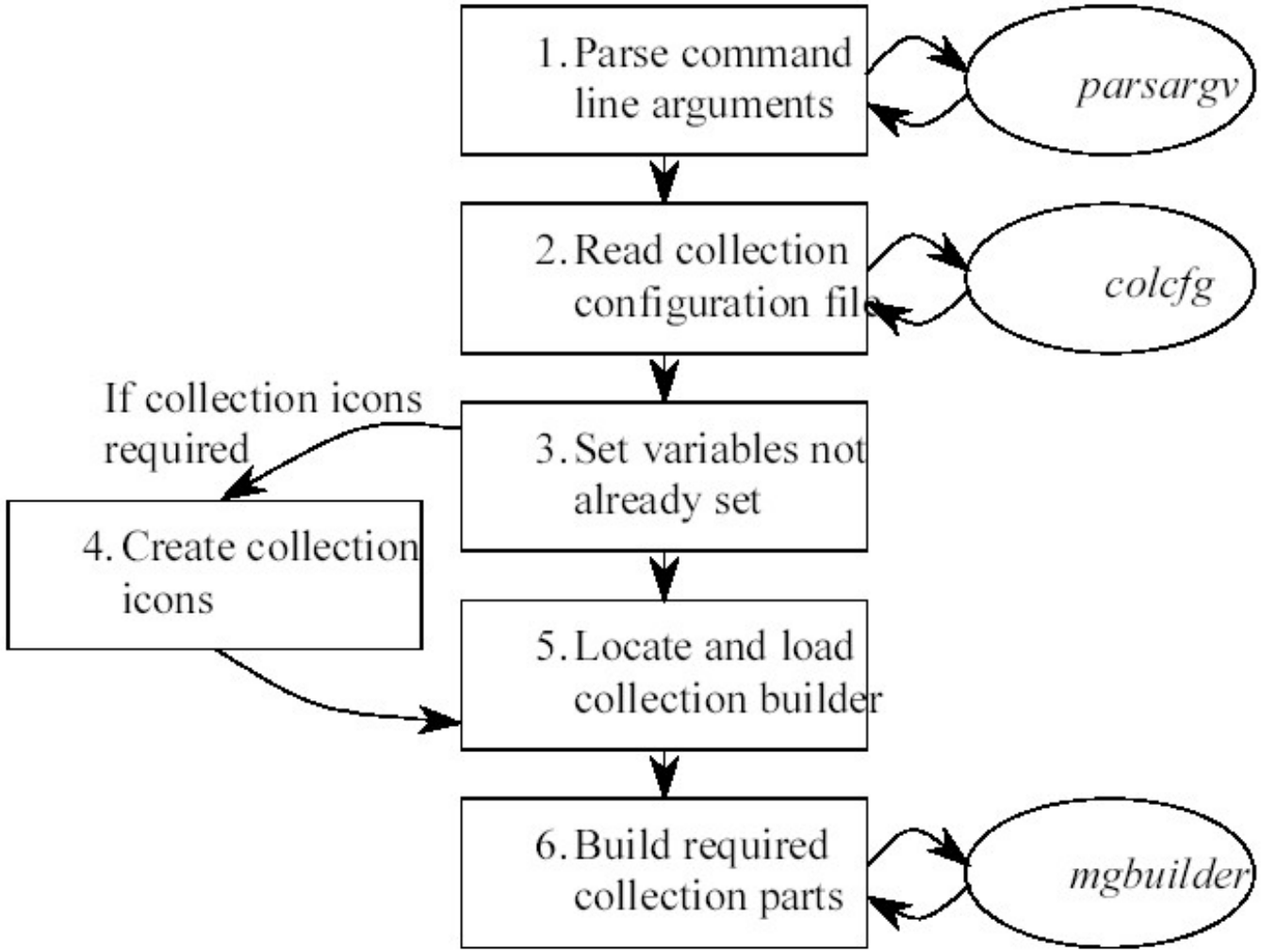
Opzioni addizionali per “import”

–importdir	Directory name	Where material to be imported is found. Defaults to *GSDLHOME/collect/col_name/import*.
–removeold	None	Remove the contents of the *archives* directory before importing.
–gzip	None	Zip up the Greenstone archive documents produced by *import* (ZIPPlug must be included in the plugin list, and *gzip* must be installed on your machine).
–groupsize	Number >0	Number of documents to group together into one Greenstone archive file, defaults 1 (that is, one document per file).
–sortmeta	Metadata tag name	Sort the documents alphabetically by the named metadata tag. However, if the collection has more than one group in the collection (i.e. *groupsize* >1), this functionality is disabled.
–OIDtype	*hash* or *incremental*	Method of creating OIDs for documents: *hash* hashes the content but is slow; *incremental* simply assigns document numbers sequentially, and is faster.

Il processo di “build”

- **Crea gli indici e le strutture dati che rendono operativa la collezione**
- **Gli indici per l’intera collezione vengono creati contemporaneamente**
  - Il processo di “build” non opera incrementalmente
  - Se si aggiunge nuovo materiale ad un archivio, bisogna ricreare l’intera collezione (ripetere il processo di “build”)

Il processo di “build”



Opzioni per “build”

–verbosity	Number 0–3	Control how much information about the process is printed to standard error; 0 gives a little, 3 gives lots.
–archivedir	Directory name	Specify where the Greenstone archive files are stored—that is, where *import.pl* puts them and where *buildcol.pl* finds them. Defaults to *GSDLHOME/collect/col_name/archives*
–maxdocs	Number >0	Indicates the maximum number of documents to be imported or built. Useful when testing a new collection configuration file, or new plugins.
–collectdir	Directory name	Specify where the collection can be found. Defaults to *GSDLHOME/collect*
–out	Filename	Specify a file to which to write all output messages, which defaults to standard error (the screen). Useful when working with debugging statements.
–keepold	None	Do not remove the result of the previous import or build operation. In the case of import, do not remove the contents of the *archives* directory; when building, do not remove the content of the *building* directory.
–debug	None	Print plugin output to standard output.

Opzioni addizionali per “build”

–builddir	Directory name	Specify where the result of building is to be stored (defaults to *GSDLHOME/collect/col_name/building*).
–index	Index name (e.g. *section:Title*)	Specify which indexes to build. This defaults to all the indexes indicated in the collection configuration file.
–allclassifications	None	Prevent the build process from removing classifications that include no documents (for example, the “X” classification in titles if there are no documents whose titles start with the letter *X*).
–create_images	None	Create collection icons automatically (to use this, GIMP, and the Gimp Perl module, must be installed).
–mode	*all*, *compress_text*, *infodb*, or *build_index*	Determine what the build process is to do (defaults to *all*). *All* does a full build, *compress_text* only compresses the document text, *infodb* creates a database of information pertaining to the collection—name, files, associated files, classification information and the like—and *build_index* builds the indexes specified in the collection configuration file or on the command line.
–no_text		Don’t store compressed text. This option is useful for minimizing the size of the built indexes if you intend always to display the original documents at run-time.

Collection Configuration File

Collection Configuration File

- **Il Collection Configuration File**
  - Definisce la struttura della collezione
  - Specifica come deve essere costruita la collezione
  - Specifica come deve essere visualizzata la collezione
- **Ogni linea del Collection Configuration File è una coppia “attributo”, “valore”**

Collection Configuration File [1/4]

creator	E-mail address of the collection's creator
maintainer	E-mail address of the collection's maintainer
public	Whether collection is to be made public or not
beta	Whether collection is beta version or not
indexes	List of indexes to build
defaultindex	The default index
subcollection	Define a subcollection based on metadata
indexsubcollections	Specify which subcollections to index
defaultsubcollection	The default indexsubcollection
languages	List of languages to build indexes in
defaultlanguage	Default index language
collectionmeta	Defines collection-level metadata
plugin	Specify a plugin to use at build time
format	A format string (explained below)
classify	Specify a classifier to use at build time

Collection configuration file [2/4]

Indici creati durante il build della collezione

Plugin da usare per convertire documenti nel formato Greenstone

Classificatore per creare una lista alfabetica di titoli

```
creator         username@email.com
maintainer      username@email.com
public          true
beta            false

indexes         section:text section:Title document:text

plugin          GAPlug
plugin          HTMLPlug -description_tags -cover_image
plugin          WordPlug -description_tags
plugin          ArcPlug
plugin          RecPlug -show_progress -use_metadata_files

classify        AZList metadata Title
```

Collection configuration file [3/4]

```
format DocumentText "<h3>[Title]</h3>\\n\\n<p>[Text]"
format DocumentImages true
format DocumentButtons "Expand Text|Expand
   Contents|Detach|Highlight"
```

Formato di presentazione dei metadati

Metadati della collezione

```
Collectionmeta collectionname "greenstone demo"
Collectionmeta collectionextra "This is a
   demonstration collection”
Collectionmeta iconcollection
   "_httpprefix_/collect/demo/images/img.gif"
```

Collection configuration file [4/4]

```
Collectionmeta collectionextra “collection description"
Collectionmeta collectionextra "This is a demonstration
   collection”
```

Descrive la collezione. Viene usato come testo nella sezione “About this collection”

```
Collectionmeta iconcollection
   "_httpprefix_/collect/demo/images/img.gif"
```

Immagine che descrive la collezione. Viene usata nella home page della collezione

Subcollections [1/4]

- Greenstone permette di costruire sotto-collezioni, e di costruire indici per ognuna di esse.
- Consideriamo una collezione costituita da documenti testuali, alcuni tratti dal “Journal of Digital Libraries” ed altri no
- Vogliamo creare due sotto-collezioni ed indici al livello di section

```
indexes     section:text
subcollection dl “Title/^Journal of Digital Libraries/i”
subcollection other “!Title/^Journal of Digital
   Libraries/i”
indexsubcollections dl other dl,other
```

Subcollections [2/4]

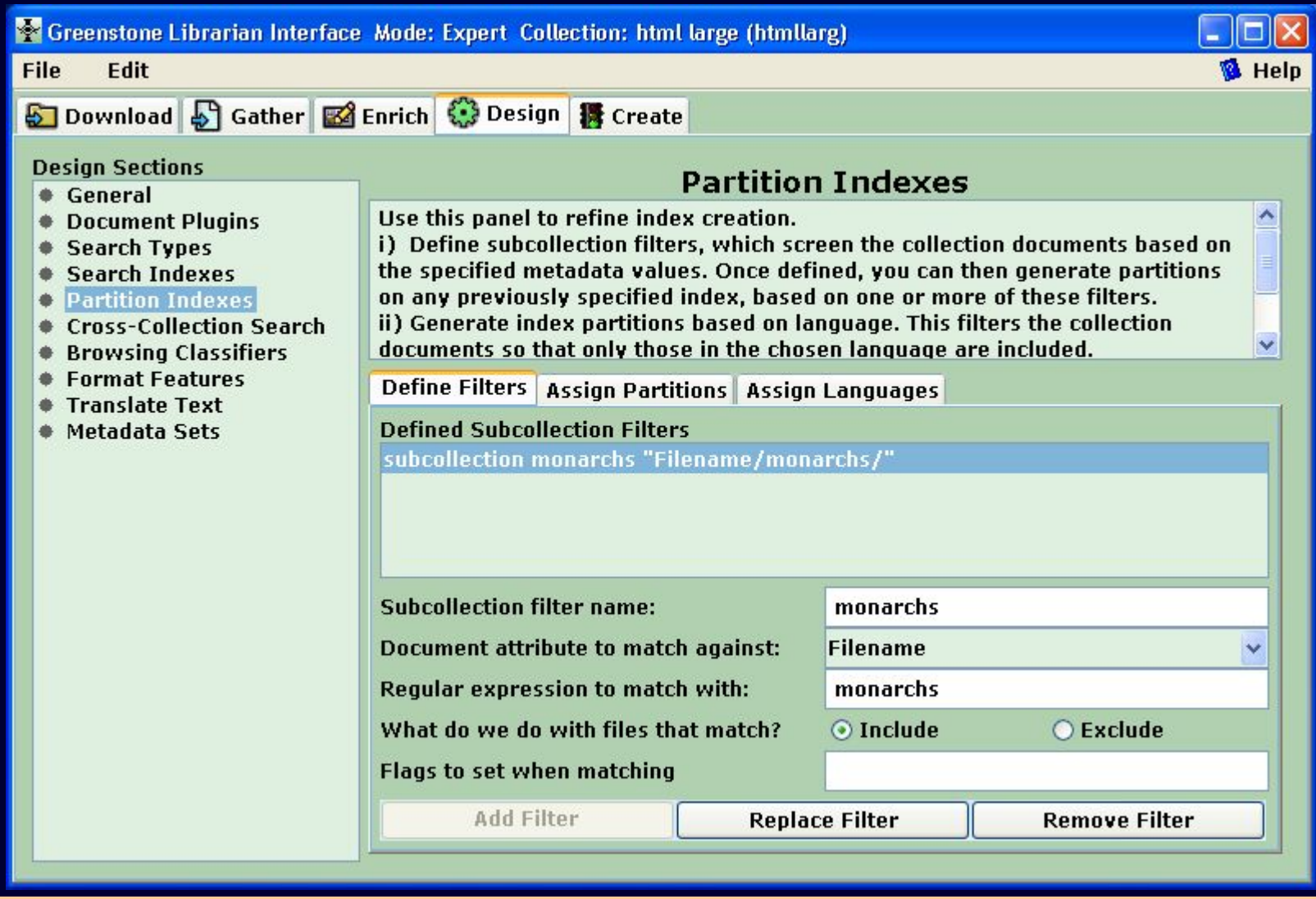
- Lo stesso meccanismo può essere utilizzato per creare indici per collezioni che contengono documenti in diverse lingue
- La lingua del documento è un metadato (en per l'inglese, it per italiano, ecc.)

```
indexes     section:text section:Title document:text
Languages it en fr
```

- Vengono creati indici separati per section text, section title, e document text per le tre diverse lingue (9 indici in totale)

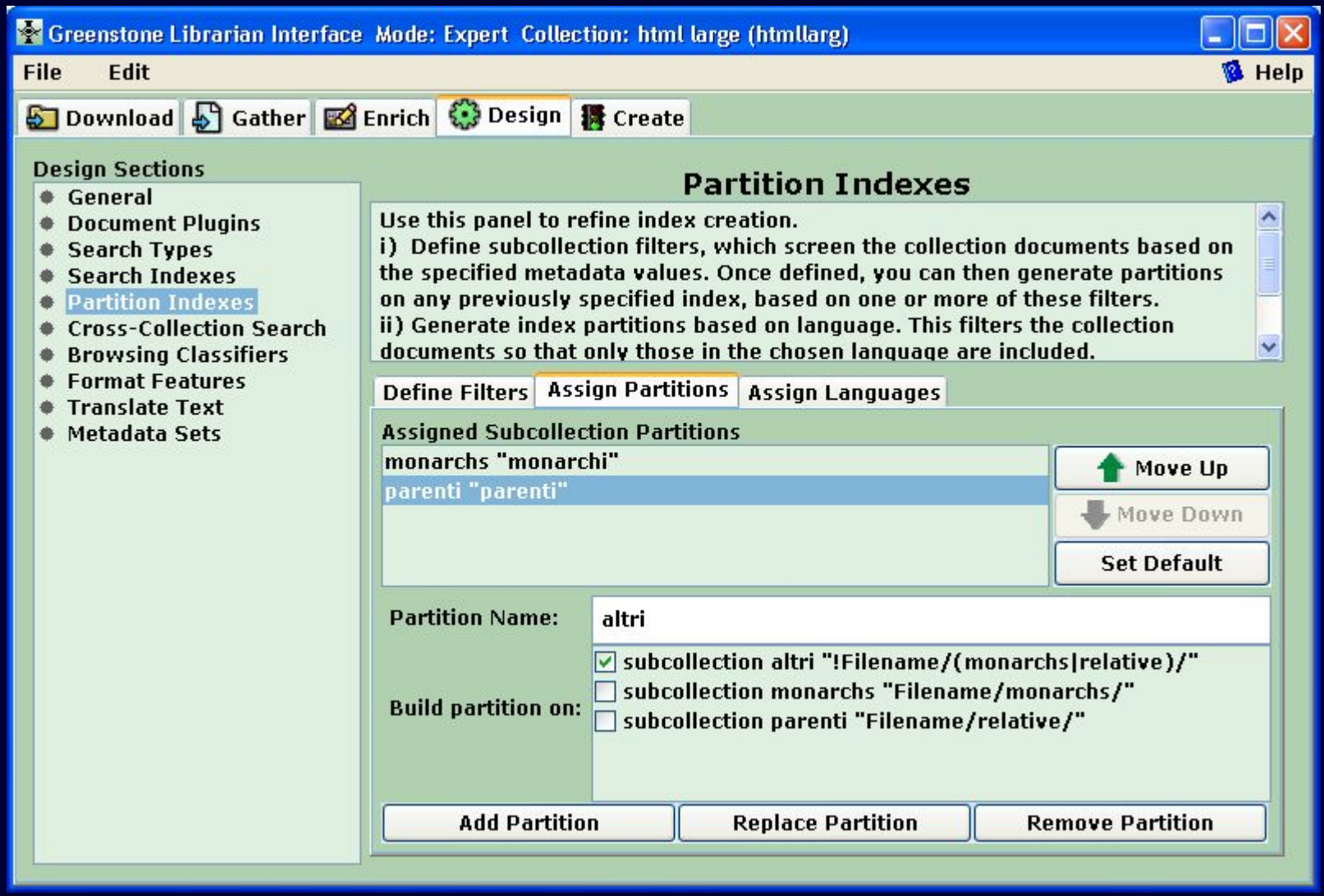
- **Definizione dei filtri**

Subcollections [3/4]



- Creazione delle sottocollezioni

Subcollections [4/4]



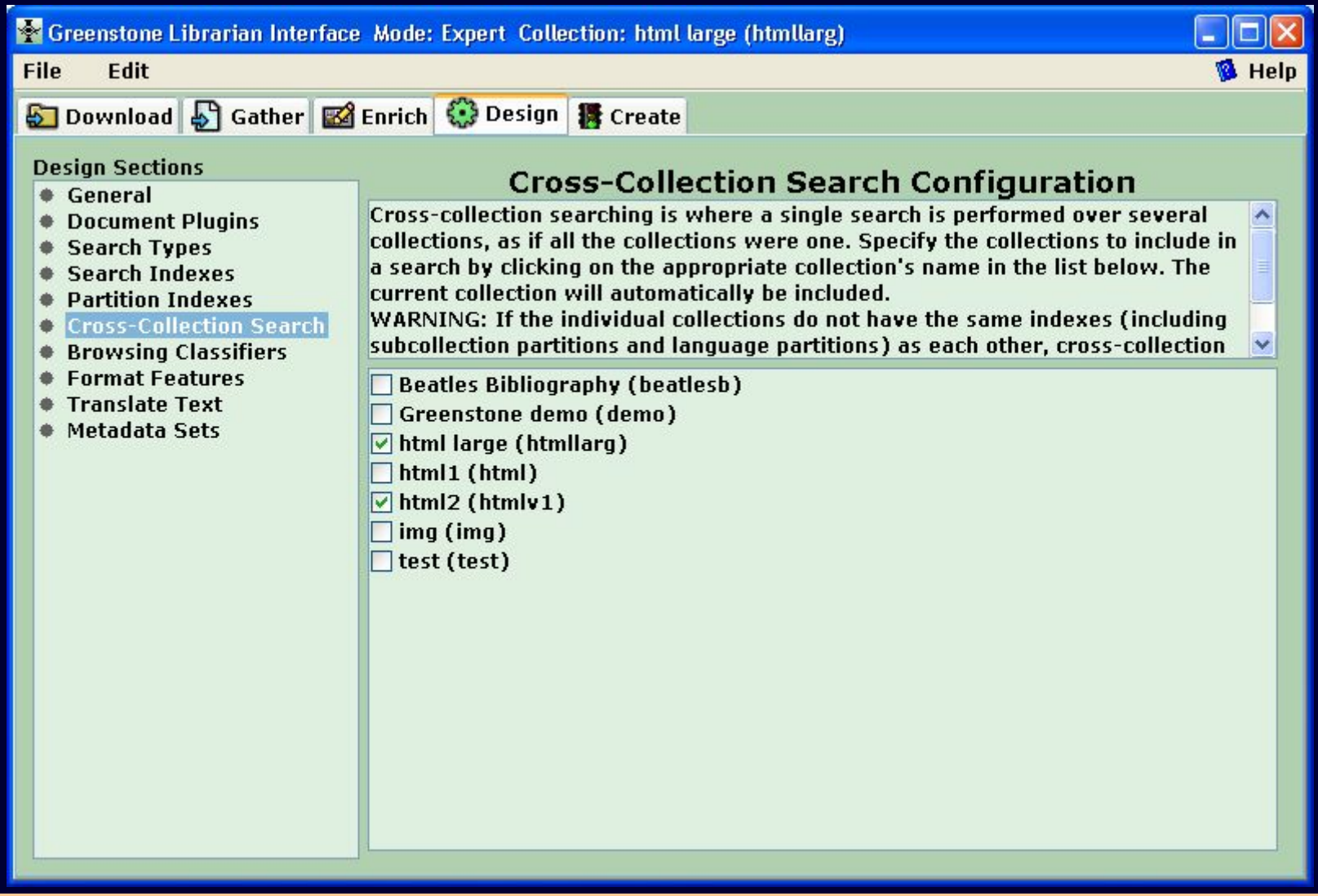
Cross-collection searching

- In Greenstone è possibile effettuare ricerche su più collezioni, come se fossero costituite da una sola collezione
- Questa funzionalità viene abilitata inserendo nel Collection Configuration File

```
supercollection  col_1 col_2 …
```

- Nel caso che le collezioni siano denominate col_1, col_2, ecc.
- Questa indicazione deve essere presente nel file di configurazione di tutte le collezioni coinvolte.

Cross-collection searching



Plug-ins

Plug-ins

- **I plug-in sono moduli software che gestiscono**
  - Conversioni di formato
  - Estrazione di metadati
- **I plug-in permettono di estendere le funzionalità di Greenstone**
  - È possibile sviluppare nuovi plug-in per estendere i tipi di documenti gestiti o i metadati che possono essere estratti
- **I plug-in sono scritti nel linguaggio Perl. Sono tutti derivati da un plug-in base: *BasPlug.***
- ***BasPlug* crea un nuovo documento archivio di Greenstone ed assegna un identificatore al documento**
- **Maggiori informazioni su ogni plug-in si possono avere digitando "perl – S pluginfo.pl nome-plugin" alla linea comandi di windows**

Plug-Ins

- **I plug-in svolgono la maggior parte del processo di “import”**
- **I diversi plug-in vengono eseguiti nell’ordine in cui compaiono nel file *collect.cfg***
  - Il file in elaborazione viene passato ai diversi plug-in, finché non se ne trova una che può elaborarlo
- **Se nessun plug-in può elaborare il file, viene generato un warning**
- **Alcuni plug-in elaborano documenti di formati diversi, mentre altri sono utilizzati come supporto al processo di importazione:**
  - *RecPlug* – elabora le directories e permette la navigazione nella struttura a directory.
  - *GAPlug* – elabora i documenti nel Greenstone Archive Format
  - *ArcPlug* – viene utilizzato durante il processo di “build” per individuare gli OID dei documenti importati (la lista si trova nel file *archives.inf* file)

Plug-ins & Document Formats

- **I plug-in sono specificati nel “collection configuration file”**
- **Il nome del file determina il formato del documento e conseguentemente il plug-in che viene utilizzato**
- **Esempi di alcuni plug-in:**

TEXTPlug
HTMLPlug
WORDPlug
PDFPlug

PSPlug
EMAILPlug
BibTexPlug
ReferPlug

SRCPlug
ImagePlug
ZIPPlug

Plug per il testo

- **TEXTPlug Plug-In**
  - *.txt
  - *.text
- **Gestisce Plain Text**
- **Crea automaticamente un metadato Title ottenuto dalla prima linea di testo del file**

HTML Plug-in

- **Opera su file del tipo**
  - *.htm, *.html, .shtml, .shm, .asp, .php, .cgi
- **Funzionalità**
  - Importa file HTML
  - Il metadato Title viene estratto dal tag HTML <title>
  - Altri metadati possono essere estratti se è presente il tag HTML <meta>
  - Riconosce ed analizza i link presenti nel file
  - I link vengono sostituiti con riferimenti al documento

Plug in per Microsoft Word Files

- **Tipi di file gestiti dal WORDPlug Plug-In**
  - *.doc
- **Importa documenti Microsoft Word**
- **Il Plug in converte file Word in HTML**

Plug in per PDF Files

- **Tipi di file gestiti dal PDFPlug Plug-In**
  - *.pdf
- **Importa file PDF (Adobe's Portable Document Format)**
- **Greenstone usa programmi indipendenti per convertire file PDF in HTML**

PostScript Files

- **PSPlug Plug-In**
  - *.ps
- **Imports PostScript Files**
- **Works best when a standard conversion program is already installed on the computer**
- **Uses simple text extraction algorithm if no conversion program is present**

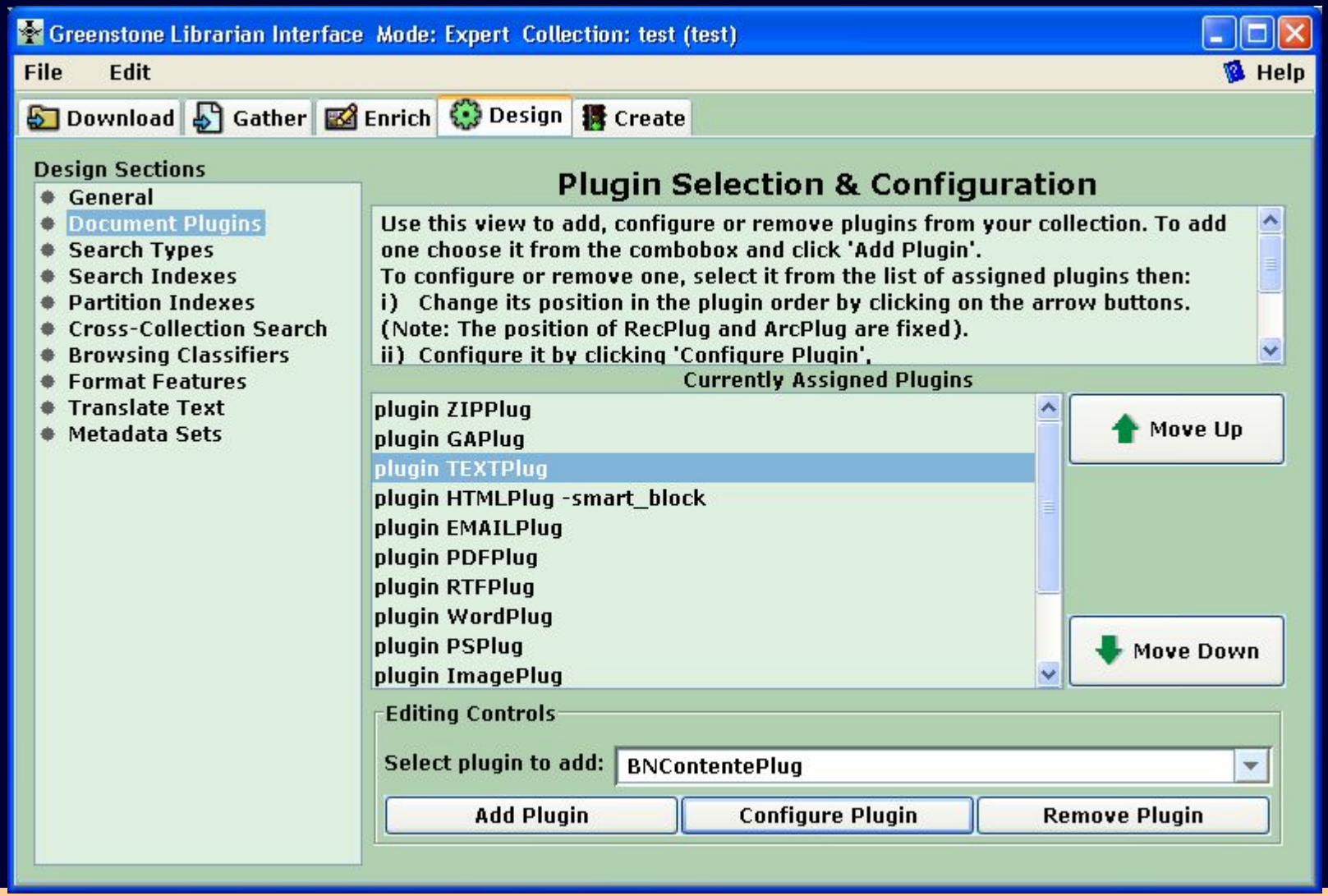
Email Files

- **EMAILPlug**
  - *.email
- **Imports files containing email**
  - Each source is checked for e-mail contents
- **Extracts metadata:**
  - Subject
  - To
  - From
  - Date
- **Deals with common formats**
  - Netscape, Eudora, Unix mail readers

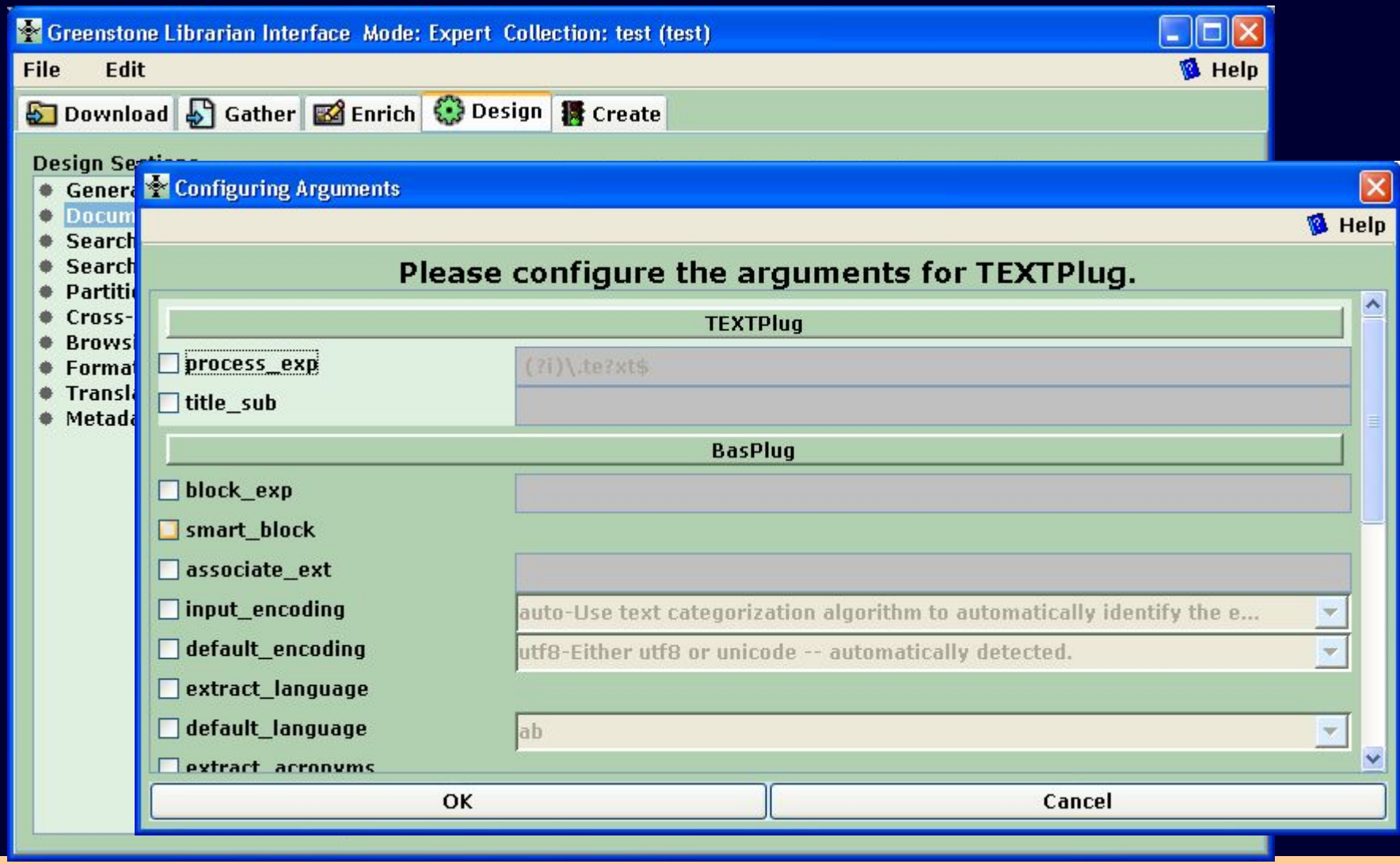
Compressed & Archived Files

- **ZIPPlug Plug-In**
  - *.zip
  - *.tar
  - .gz
  - *.z
  - *.tgz
  - *.bz
- **Relies on standard utility programs being present**

Configurazione Plug-in con GLI



Configurazione Plug-in con GLI



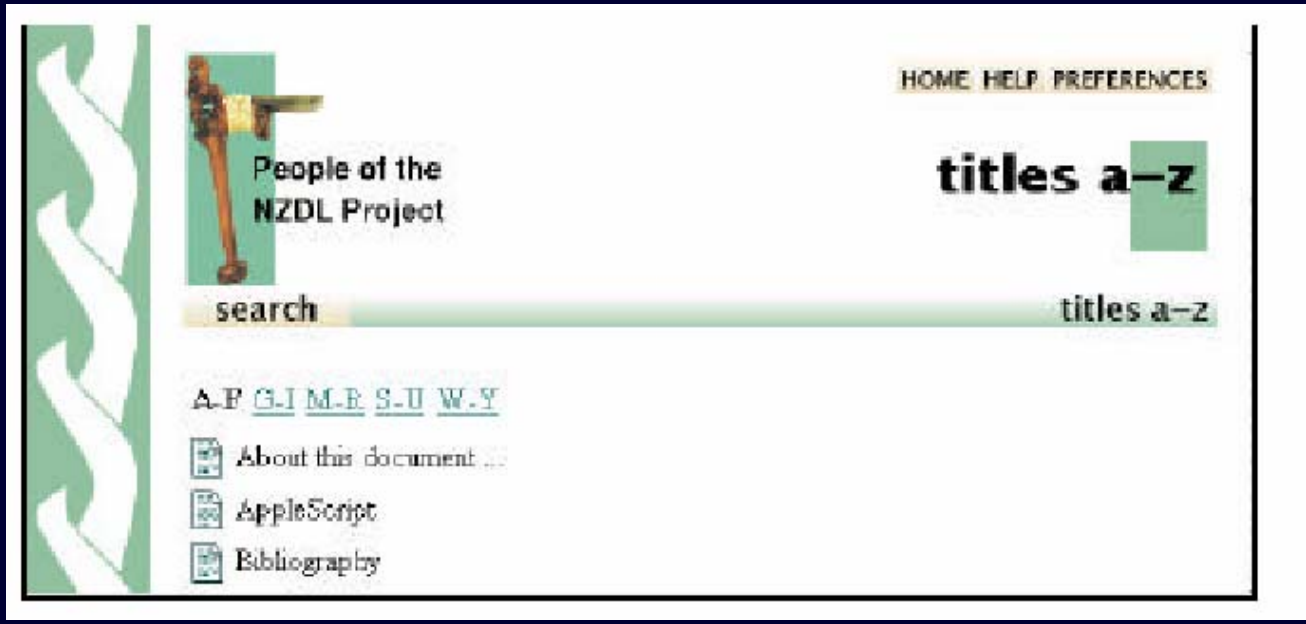
Classifiers

I Classifiers

- **Gestiscono strutture per il browsing della collezione**
- **Vengono specificati nel Collection Configuration File**
- **Per ogni classifier vi è una linea del tipo**
  - `classify nome_classifier opzioni`
- **I programmatori possono scrivere nuovi classifiers per creare nuove strutture di browsing**

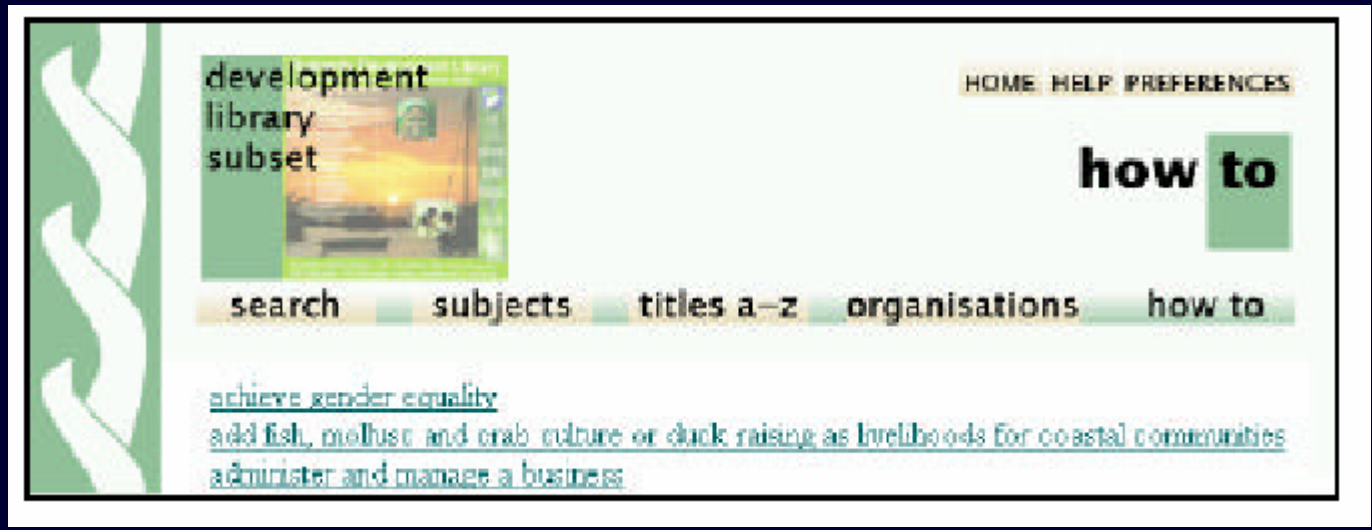
Esempi di Classifier [1/4]

- **AZList classifier**
  - Crea una lista ordinata alfabeticamente di elementi
  - Ad es. `Classify AZList –metadata Title`



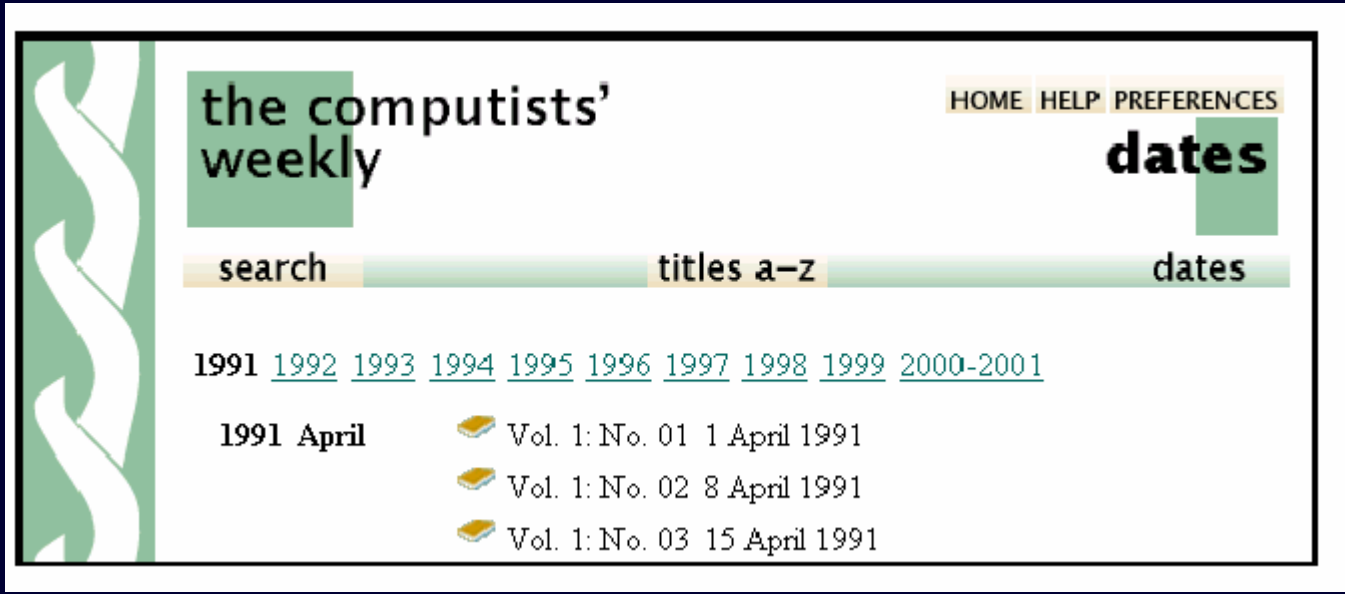
Esempi di Classifier [2/4]

- **List classifier**
  - Crea una lista ordinata di elementi e li visualizza senza alcun ordine specifico
  - Ad es. `classify List –metadata Howto`



Esempi di Classifier [3/4]

- **DateList classifier**
  - Crea una lista ordinata di elementi data
  - Ad es. `classify DateList –metadata date`



Esempi di Classifier [4/4]

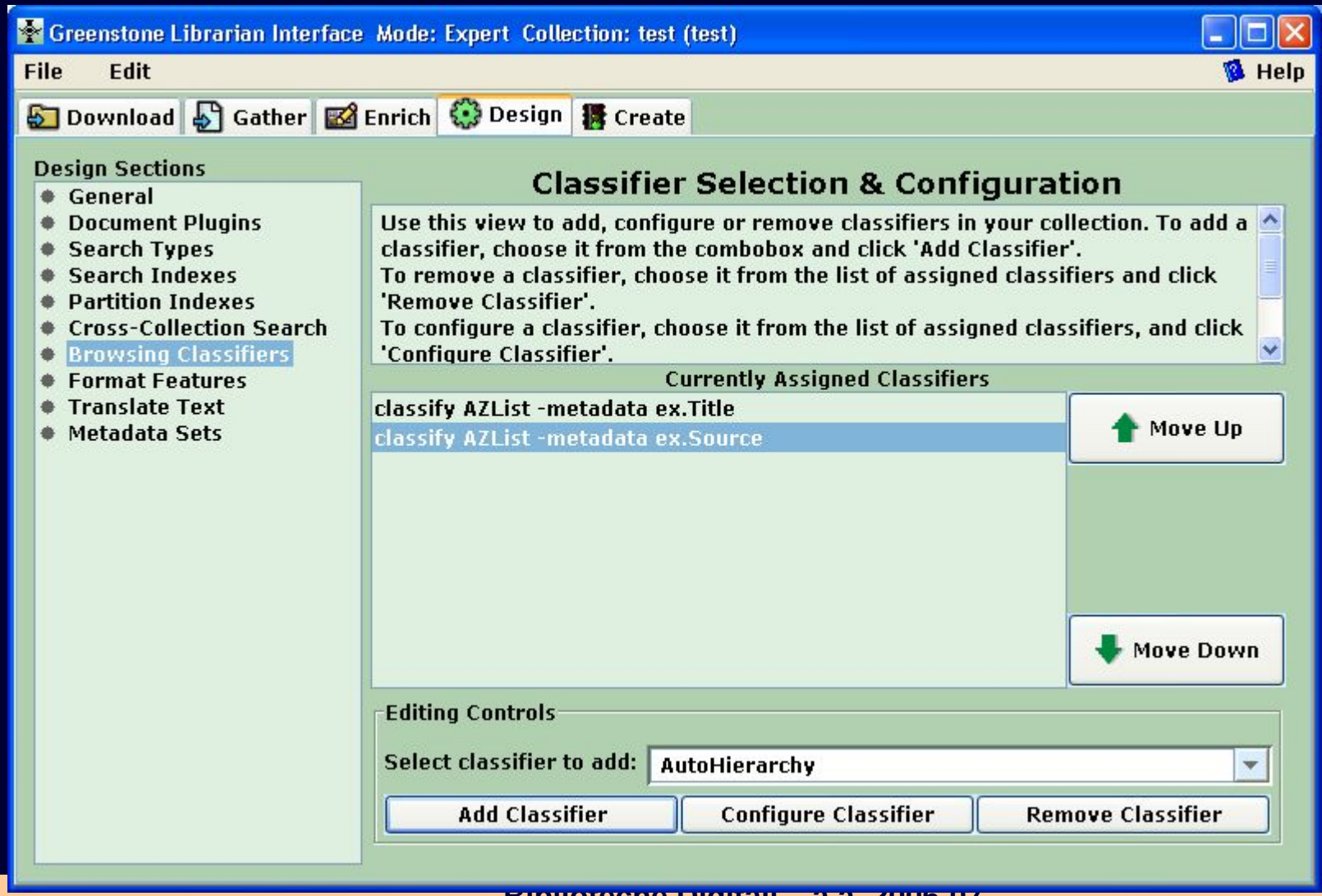
- **Classifier gerarchici**
  - Creano classificazioni gerarchiche e sono utili per la classificazione di soggetti ed organizzazioni
  - Ad es. `classify Hierarchy –hfile sub.txt –metadata Subject –sort Title`

I classifiers

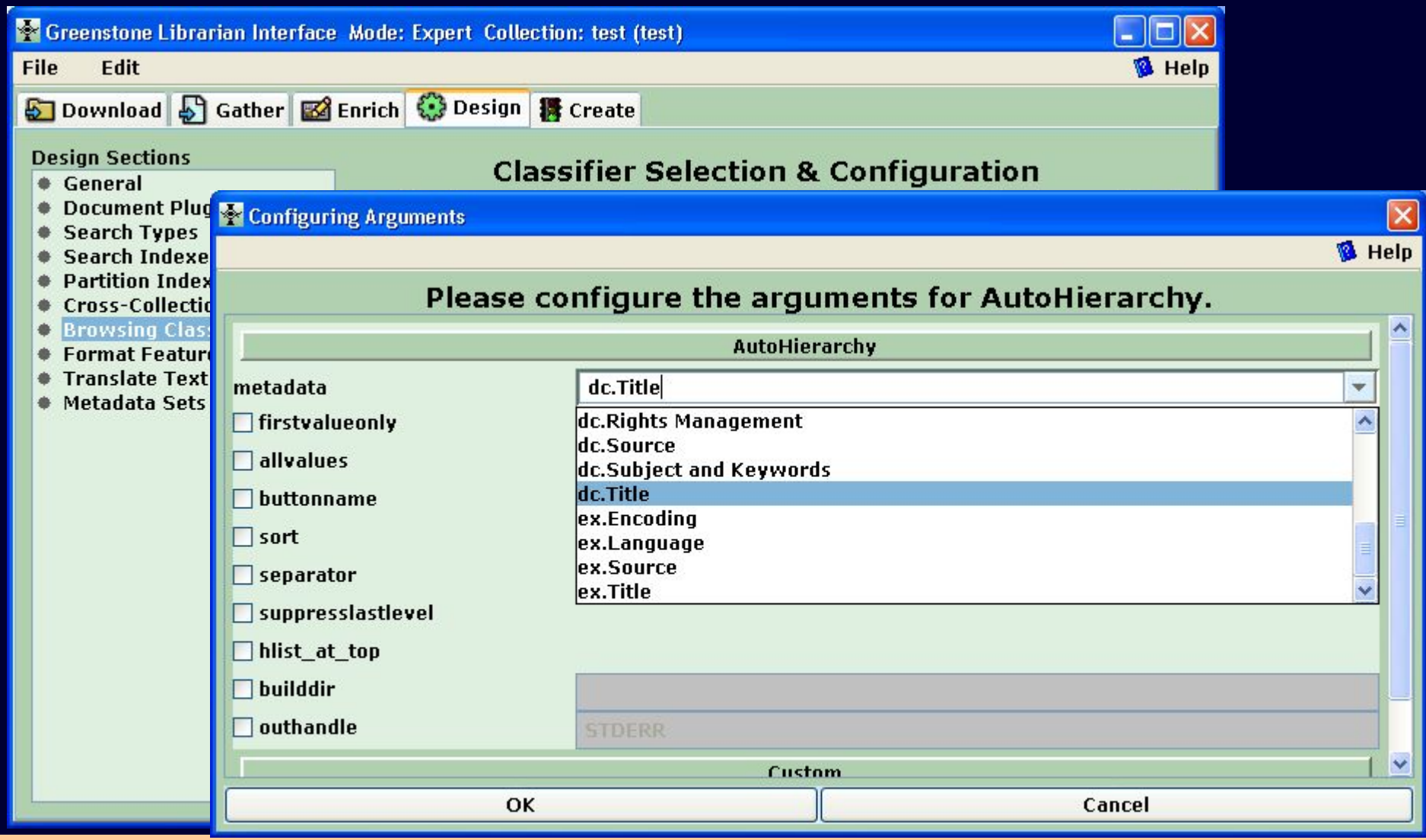
- **Informazioni sui classifiers si possono avere digitando dalla linea comandi**
  - `perl – S classinfo.pl nome-classifier`

Hierarchy		Hierarchical classification
	hfile	Classification file
	metadata	Metadata element to test against *hfile* identifier
	sort	Metadata element used to sort documents within leaves (defaults to *Title*)
	buttonname	Name of the button used to access this classifier (defaults to value of metadata argument)
List		Alphabetic list of documents
	metadata	Include documents containing this metadata element
	buttonname	Name of button used to access this classifier (defaults to value of metadata argument)
SectionList		List of sections in documents
AZList		List of documents split into alphabetical ranges
	metadata	Include all documents containing this metadata element
	buttonname	Name of button used to access this classifier (defaults to value of metadata argument)
AZSectionList		Like *AZList* but includes every section of the document
DateList		Similar to *AZList* but sorted by date

Gestione del Classifiers con la GLI



Gestione del Classifiers con la GLI



Indici

Uso di indici per la ricerca

- **La ricerca è resa possibile da indici costruiti sulle diverse componenti dei documenti**
  - Documenti intero
  - Paragrafi
  - Titoli
  - Sezioni
  - Titoli di sezione
  - Titoli delle figure
  - Ecc.

Indici

- **Gli indici possono essere creati automaticamente utilizzando**
  - I documenti
  - File di supporto che contengono i valori dei metadati
- **Gli indici devono essere ricostruiti automaticamente**
  - Quando un nuovo documento viene inserito nella collezione

Plug-ins per gli indici

- **I documenti sono convertiti in formato XML standard da plug-in specifici. Queste rappresentazioni XML dei documenti vengono utilizzate per l'indicizzazione**

- **DTD del Metadata file**

```
<!DOCTYPE GreenstoneDirectoryMetadata [
<!ELEMENT DirectoryMetadata (FileSet*)>
<!ELEMENT FileSet (FileName+,Description)>
<!ELEMENT FileName (#PCDATA)>
<!ELEMENT Description (Metadata*)>
<!ELEMENT Metadata (#PCDATA)>
<ATTLIST Metadata name CDATA #REQUIRED>
<ATTLIST Metadata mode (accumulate|override) "override">
]>
```

Esempio di XML Metadata File

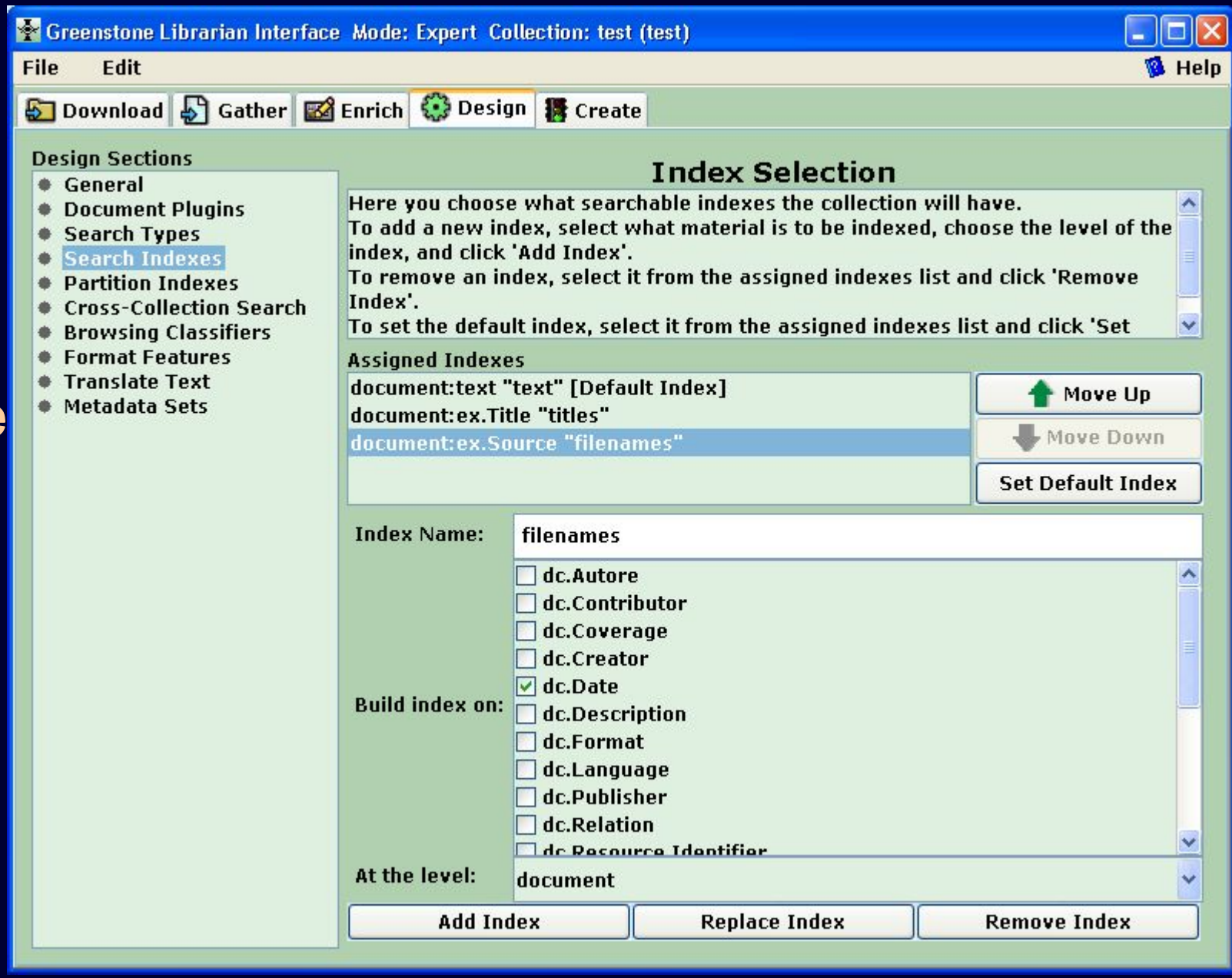
```
<?xml version="1.0" ?>
<!DOCTYPE GreenstoneDirectoryMetadata SYSTEM
"http://greenstone.org/dtd/GreenstoneDirectoryMetadata/1.0/GreenstoneDirectoryM
etadata.dtd">
<DirectoryMetadata>
<FileSet>
<FileName>nugget.*</FileName>
<Description>
<Metadata name="Title">Nugget Point Lighthouse</Metadata>
<Metadata name="Place" mode="accumulate">Nugget Point</Metadata>
</Description>
</FileSet>
<FileSet>
<FileName>nugget-point-1.jpg</FileName>
<Description>
<Metadata name="Title">Nugget Point Lighthouse</Metadata>
<Metadata name="Subject">Lighthouse</Metadata>
</Description>
</FileSet>
</DirectoryMetadata>
```

Tagging Document Files

- **Una diversa modalità per associare metadati sui quali creare gli indici, consiste nell'aggiungere dei metadati direttamente nei documenti**

```
<!--
<Section>
<Description>
<Metadata name="Title"> Realizing human rights for poor
people: Strategies for achieving the international
development targets </Metadata>
</Description>
-->
(text of section goes here)
<!--
</Section>
-->
```

Gestione degli indici con la GLI



Come formattare l’output

Introduzione

- **Le pagine web visualizzate da Greenstone non sono preesistenti ma vengono generate**
- **Le modalità di visualizzazione sono controllate dal comando “format” del Collection Configuration File**
- **Elementi della pagina controllabili**
  - Item della pagina che presentano i documenti
  - Liste prodotte dai classifiers e risultati delle ricerche

Visualizzazione degli item nella pagina

format DocumentImages true/false	If *true*, display a cover image at the top left of the document page (default *false*).
format DocumentHeading formatstring	If *DocumentImages* is *false*, the format string controls how the document header shown at the top left of the document page looks (default *[Title]*).
format DocumentContents true/false	Display table of contents (if document is hierarchical), or next/previous section arrows and "page k of n" text (if not).
format DocumentButtons string	Controls the buttons that are displayed on a document page (default *Detach|Highlight*).
format DocumentText formatstring	Format of the text to be displayed on a document page: default * <table width=537>* *<tr><td>[Text]</td></tr>* *</table> *
format DocumentArrowsBottom true/false	Display next/previous section arrows at bottom of document page (default *true*).
format DocumentUseHTML true/false	If *true*, each document is displayed inside a separate frame. The Preferences page will also change slightly, adding options applicable to a collection of HTML documents, including the ability to go directly to the original source document (anywhere on the Web) rather than to the Greenstone copy.

Come formattare le liste

- **`Format lista-parte comandi`**
  - La prima parte (`list`) è obbligatoria ed identifica le liste alle quali applicare i comandi di formattazione
  - Search è la lista generata da una ricerca, mentre CL1, CL2, … sono le liste generate dal primo, secondo, … classificatore
  - La seconda parte (`parte`) è opzionale e specifica a quale parte della lista i comandi vanno applicati (HList, VList, DateList)
    - ➔ **Ad es. `format CL4Vlist` si applica a tutte le VList in CL4**

Come formattare le liste

- **`Comandi` è una stringa che specifica come formattare la lista**
- **Può contenere codice HTML, metadati ed i seguenti elementi**

[Text]	The document's text
[link] ... [/link]	The HTML to link to the document itself
[icon]	An appropriate icon (e.g. the little text icon in a *Search Results* string)
[num]	The document number (useful for debugging).
[metadata-name]	The value of this metadata element for the document, e.g. *[Title]*

Esempio [1/8]

- **Esempio di classifiers e format commands della demo collection**

```
1  classify Hierarchy -hfile sub.txt -metadata Subject -sort Title
2  classify AZList    -metadata Title
3  classify Hierarchy -hfile org.txt -metadata Organisation -sort Title
4  classify List      -metadata Howto
5  format SearchVList "<td valign=top [link][icon][/link]</td><td>{If}
                       {[parent(All':'):Title],[parent(All':'):Title]:}
6                      [link][Title][/link]</td>"
7  format CL4Vlist          "<br>[link][Howto][/link]"
8  format DocumentImages    true
9  format DocumentText      "<h3>[Title]</h3>\\n\\n<p>[Text]"
```

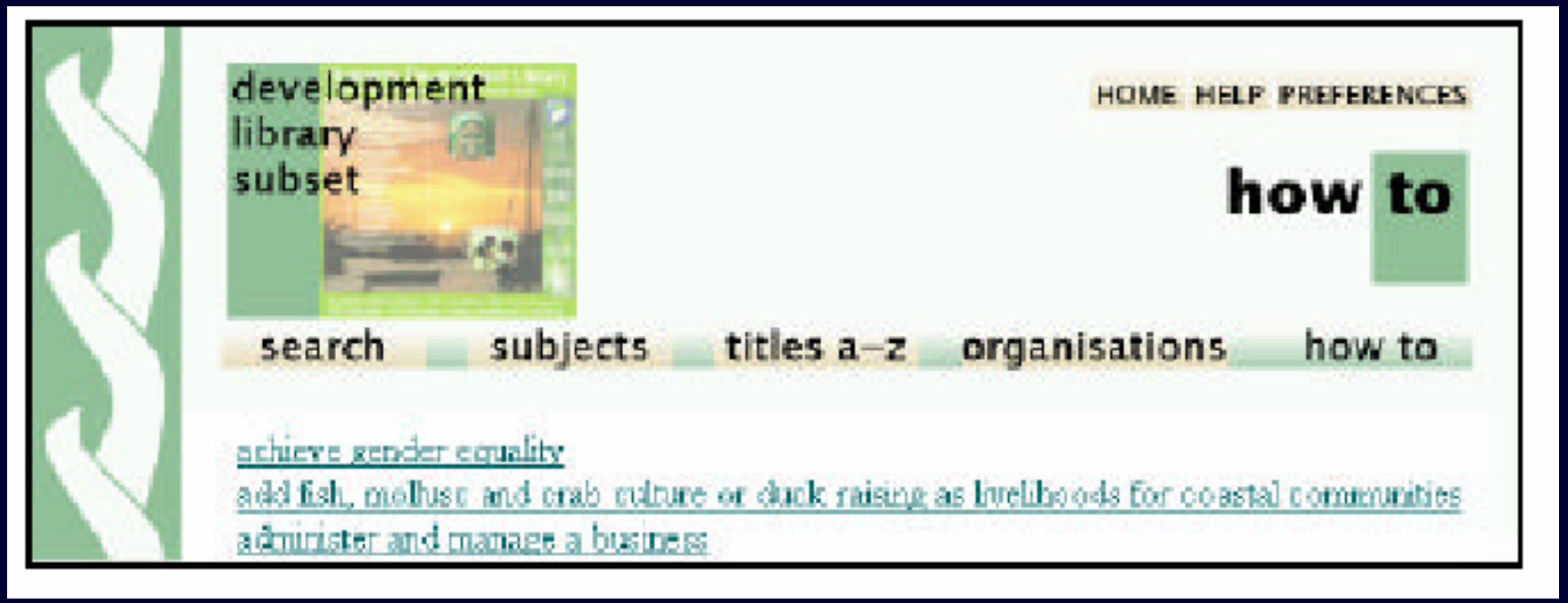
Esempio [2/8]

Howto classifier. È il quarto classifier (CL4)
È un List classifier che genera una lista di titoli di documenti

```
1  classify Hierarchy -hfile sub.txt -metadata Subject -sort Title
2  classify AZList    -metadata Title
3  classify Hierarchy -hfile org.txt -metadata Organisation -sort Title
4  classify List      -metadata Howto
5  format SearchVList "<td valign=top [link][icon][/link]</td><td>{If}
                       {[parent(All':'):Title],[parent(All':'):Title]:}
6                      [link][Title][/link]</td>"
7  format CL4Vlist        "<br>[link][Howto][/link]"
8  format DocumentImages  true
9  format DocumentText    "<h3>[Title]</h3>\\n\\n<p>[Text]"
```

Comando di formattazione di CL4
Gli elementi figlio degli elementi top-level sono visualizzati come una VList
Ogni elemento si trova su una nuova linea e contiene il testo del campo Howto collegato al documento

Esempio [3/8]


development
library
subset
HOME HELP PREFERENCES
how to
search
subjects
titles a–z
organisations
how to
achieve gender equality
add fish, mollusc and crab culture or duck raising as livelihoods for coastal communities
administer and manage a business

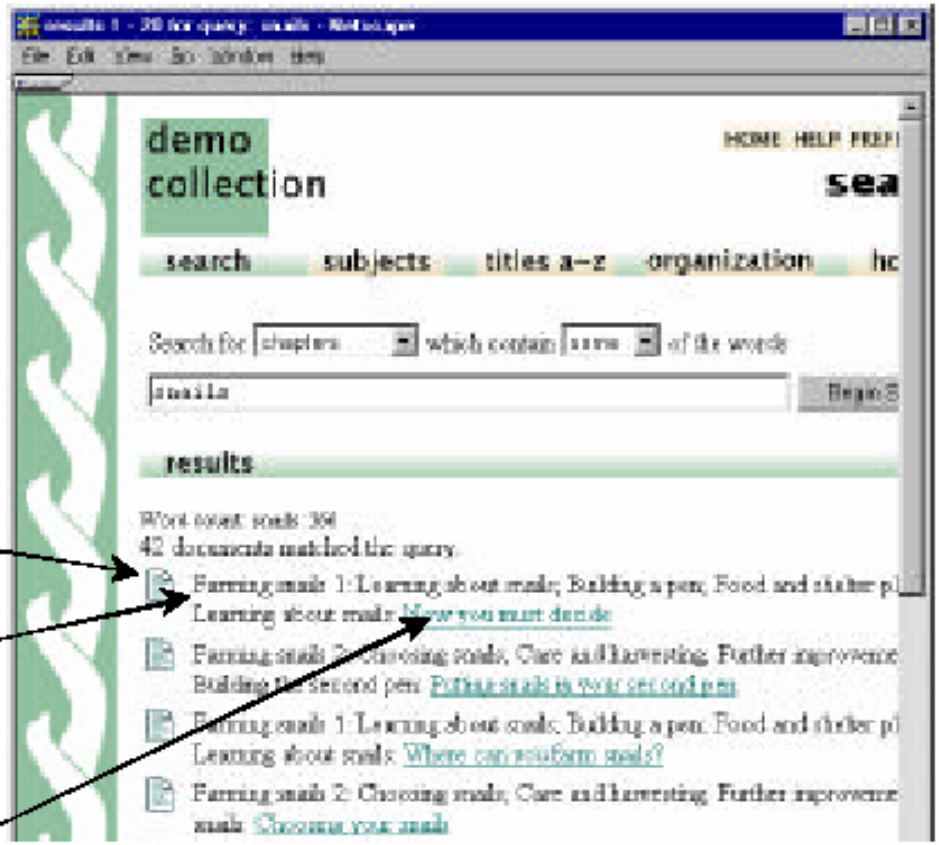
Esempio [4/8]


```
format DocumentImages true
```

```
format DocumentText
   "<h3>[Title]</h3>\\n\\n<p>[Text]"
```

```
1  classify Hierarchy
2  classify AZList
3  classify Hierarchy
4  classify List
5  format SearchVList "<td valign=top [link][icon][/link]</td><td>{If}
                       {[parent(All':'):Title],[parent(All':'):Title]:}
                       [link][Title][/link]</td>"
6
7  format CL4Vlist          "<br>[link][Howto][/link]"
8  format DocumentImages  true
9  format DocumentText    "<h3>[Title]</h3>\\n\\n<p>[Text]"
```

Esempio [5/8]



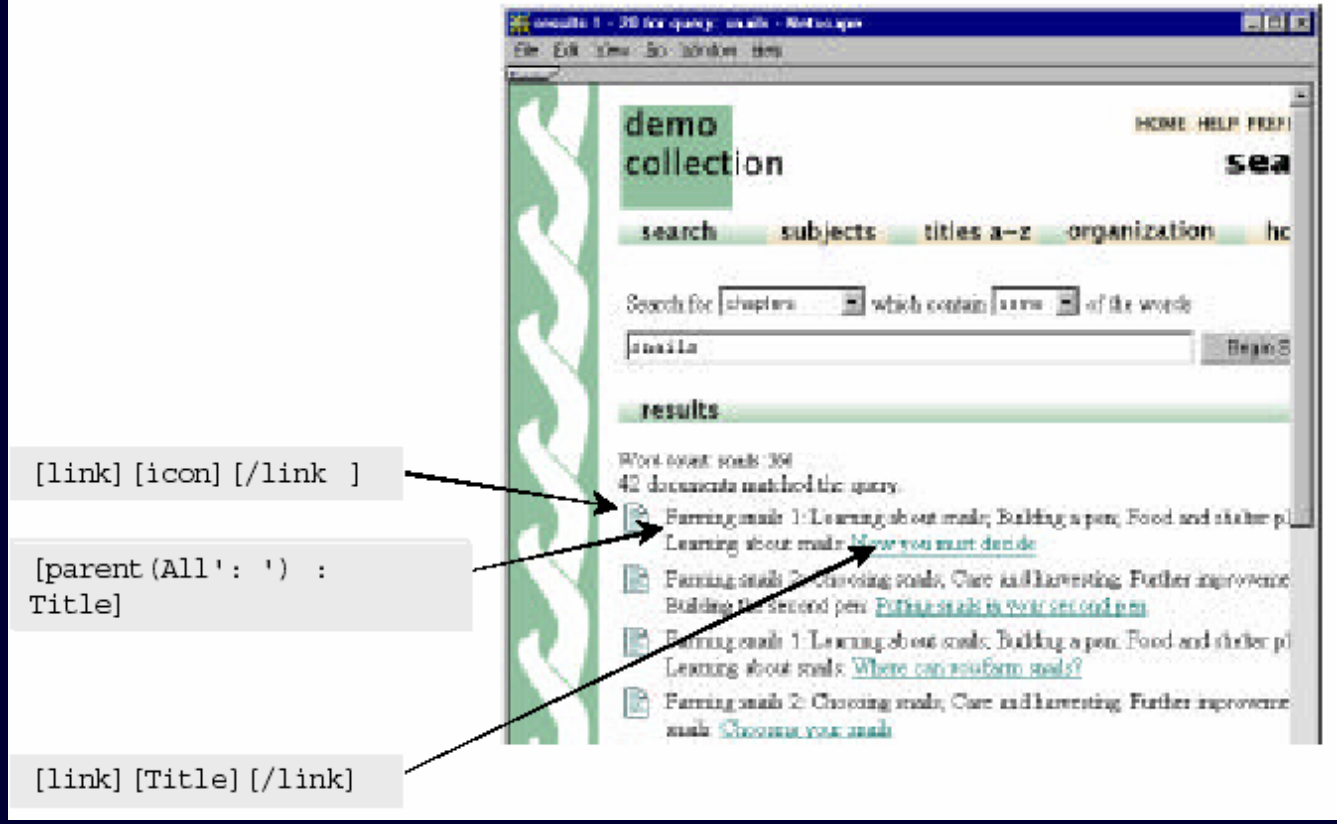
[link] [icon] [/link]

[parent(All': ') : Title]

[link] [Title] [/link]

```
1  classify Hie
2  classify AZI
3  classify Hie
4  classify List          -metadata Howto
5  format SearchVList "<td valign=top [link] [icon] [/link]</td><td>{If}
                       {[parent(All':'):Title],[parent(All':'):Title]:}
6                      [link] [Title] [/link]</td>"
7  format CL4Vlist           "<br>[link] [Howto] [/link]"
8  format DocumentImages     true
9  format DocumentText       "<h3>[Title]</h3>\\n\\n<p>[Text]"
```

Esempio [6/8]



- Una versione semplice per il formato del classificatore Howto dovrebbe essere del tipo

```
<td valign=top>[link][icon][/link]</td>
<td>[link][Title][/link]</td>
```

- In questo modo si ha un link al documento tramite la sua icona ed un link al documento tramite il titolo

- I documenti della collezione hanno una struttura gerarchica (book, section, subsection, ecc.)

- La search dà come risultato una specifica parte del documento, per cui con

```
<td>[link][Title][/link]</td>
```

visualizzo solo il titolo della componente trovata. Se voglio visualizzare tutta la struttura gerarchica di titoli devo utilizzare un elemento specifico (*parent*) che fornisce il 'parent' di un oggetto o, se si specifica 'All', fornisce tutta la struttura.

```
<td>{[parent(All' : '):Title]: }[link][Title][/link]</td>
```

Questa stringa genera tutti i titoli a partire da Book, separati da ':'. Rimane il problema di un documento che non ha struttura. In tal caso parent è una stringa vuota, per cui avrei come risultato

```
: Titolo del documento
```

- Per evitare questo inconveniente utilizzo uno statement *if*

  `{If} {[metadata], se-non-vuoto, se-vuoto}`

  verifica se il valore in [metadata] è vuoto o no, ed esegue le azioni corrispondenti

- Lo statement *or*

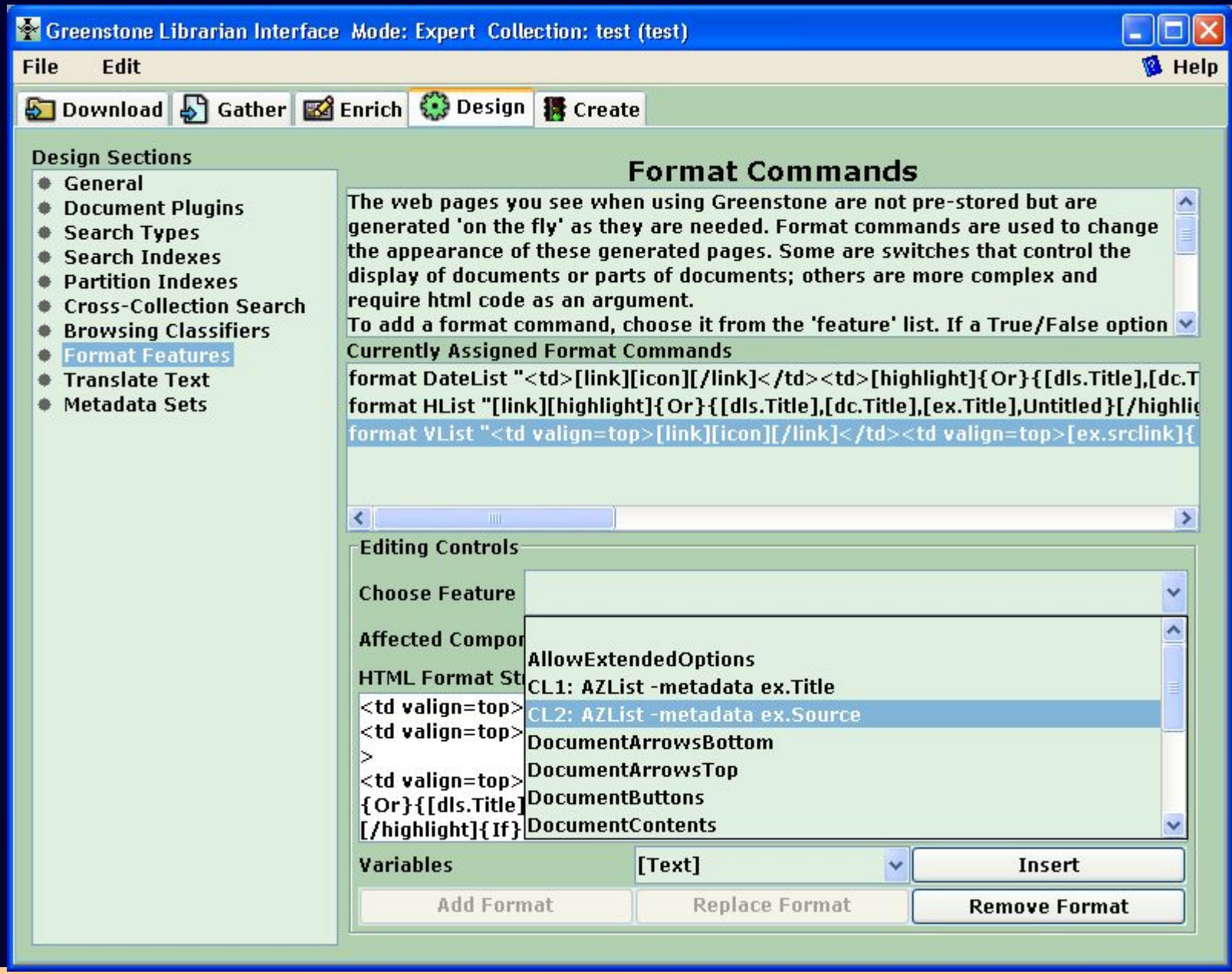
  `{Or} {azione-1, else azione-2, else azione-3, ecc.}`

  valuta tutte le azioni in sequenza, finché non ne trova una che non generi una stringa vuota.

- Quindi il format corretto risulta essere

```
<td valign=top>[link][icon][/link]</td>
<td> {If} {[parent(All' : '):Title],
           [parent(All' : '):Title]:}
     [link][Title][/link]</td>
```

Format features



Riferimenti

- **Greenstone Developer Guide, cap. 2**
  http://prdownloads.sourceforge.net/greenstone/Develop-en.pdf