Approach and Software Design

**For Data Extractor**

Contents

[System architecture 2](#_Toc311230959)

[Class model 2](#_Toc311230960)

[Dynamic model 3](#_Toc311230961)

[External libraries 4](#_Toc311230962)

[Approach on file formats 5](#_Toc311230963)

[MS Office 97-2003 5](#_Toc311230964)

[Retrieving meta info 5](#_Toc311230965)

[Retrieving content text 7](#_Toc311230966)

[MS Office 2007 8](#_Toc311230967)

[Retrieving meta info 8](#_Toc311230968)

[Retrieving content text 10](#_Toc311230969)

[MS RTF 11](#_Toc311230970)

[Text 11](#_Toc311230971)

[Adobe PDF 11](#_Toc311230972)

[Retrieving meta info 12](#_Toc311230973)

[Retrieving content text 13](#_Toc311230974)

[JPEG, TIFF 13](#_Toc311230975)

# System architecture

## Class model

This system has been composed with following classes in the static model:

* CDEFile class

An abstract class to generalize files involving various meta information.

* CDEFolder

An abstract class to generalize folders including file.

* COfficeFile

An abstract class to generalize the Microsoft Office 2003 and 2007 files (Word, Excel, PowerPoint).

* CPdfFile

A class to generalize the Adobe PDF file.

* CRtfFile

A class to generalize the Microsoft RTF file.

* CTxtFile

A class to generalize the text file.

* CDocFile, CPptFile, CXlsFile, CDocxFile, CPptxFile, CXlsxFile

Classes to generalize individual Microsoft Office files.

* CImageFile

An abstract class to generalize various image files.

* CTiffFile, CJpegFile, CBmpFile

Classes to generalize individual image files.



Figure1. class diagram

## Dynamic model

This system starts with an argument of target directory and search all data files recursively included in the target directory and extract the meta and contents data from each file. Finally serialize all extracted data into an JSON string and save to a destination file.

****

Figure2. sequence diagram for main flow

The figure shows the main flow of the program which carries out searching of data files, extracting individual meta data and contents, reporting in the form of JSON string.

## External libraries

This system includes various external libraries necessary for parsing of data file formats.



Figure3. external libraries and dependencies

The figure shows the module composition of main program and external libraries and the dependencies among them.

# Approach on file formats

## MS Office 97-2003

### Retrieving meta info

The system uses the same method to extract meta information for every MS Office 97-2003 files. The files have a structure of *compound file* which is storage and stream standard based on Microsoft COM. *IPropertySetStorage* interface has a method *Open* that fetch various properties from 3 types of standard property set format Identifiers:

<http://msdn.microsoft.com/en-us/library/windows/desktop/aa380060(v=VS.85).aspx>

* From DOC file

Author name: FMTID\_SummaryInformation -> PIDSI\_AUTHOR

Title: FMTID\_SummaryInformation -> PIDSI\_TITLE

Subject: FMTID\_SummaryInformation -> PIDSI\_SUBJECT

Keywords: FMTID\_SummaryInformation -> PIDSI\_KEYWORDS

Status: FMTID\_SummaryInformation -> PIDSI\_APPNAME

Company: FMTID\_DocSummaryInformation -> PIDDSI\_COMPANY

Application:

Comments: FMTID\_SummaryInformation -> PIDSI\_COMMENTS

Template used: FMTID\_SummaryInformation -> PIDSI\_TEMPLATE

Revision number: FMTID\_SummaryInformation -> PIDSI\_REVNUMBER

Total editing time: FMTID\_SummaryInformation -> PIDSI\_EDITTIME

Last saved: FMTID\_SummaryInformation -> PIDSI\_LASTSAVE\_DTM

Last edited by: FMTID\_SummaryInformation -> PIDSI\_LASTAUTHOR

Last printed: FMTID\_SummaryInformation -> PIDSI\_LASTPRINTED

Recent hyperlinks list:

Page count: FMTID\_SummaryInformation -> PIDSI\_PAGECOUNT

Word count: FMTID\_SummaryInformation -> PIDSI\_WORDCOUNT

Character count: FMTID\_SummaryInformation -> PIDSI\_CHARCOUNT

Track changes:

Slide notes:

GUID:

* From PPT file

Author name: FMTID\_SummaryInformation -> PIDSI\_AUTHOR

Title: FMTID\_SummaryInformation -> PIDSI\_TITLE

Subject: FMTID\_SummaryInformation -> PIDSI\_SUBJECT

Keywords: FMTID\_SummaryInformation -> PIDSI\_KEYWORDS

Status: FMTID\_SummaryInformation -> PIDSI\_APPNAME

Company: FMTID\_DocSummaryInformation -> PIDDSI\_COMPANY

Application:

Comments: FMTID\_SummaryInformation -> PIDSI\_COMMENTS

Template used: FMTID\_SummaryInformation -> PIDSI\_TEMPLATE

Revision number: FMTID\_SummaryInformation -> PIDSI\_REVNUMBER

Total editing time: FMTID\_SummaryInformation -> PIDSI\_EDITTIME

Last saved: FMTID\_SummaryInformation -> PIDSI\_LASTSAVE\_DTM

Last edited by: FMTID\_SummaryInformation -> PIDSI\_LASTAUTHOR

Last printed: FMTID\_SummaryInformation -> PIDSI\_LASTPRINTED

Recent hyperlinks list:

Page count: FMTID\_DocSummaryInformation -> PIDDSI\_SLIDECOUNT

Word count: FMTID\_SummaryInformation -> PIDSI\_WORDCOUNT

Character count: FMTID\_SummaryInformation -> PIDSI\_CHARCOUNT

Track changes:

Slide notes:

GUID:

* From XLS file

Author name: FMTID\_SummaryInformation -> PIDSI\_AUTHOR

Title: FMTID\_SummaryInformation -> PIDSI\_TITLE

Subject: FMTID\_SummaryInformation -> PIDSI\_SUBJECT

Keywords: FMTID\_SummaryInformation -> PIDSI\_KEYWORDS

Status: FMTID\_SummaryInformation -> PIDSI\_APPNAME

Company: FMTID\_DocSummaryInformation -> PIDDSI\_COMPANY

Application:

Comments: FMTID\_SummaryInformation -> PIDSI\_COMMENTS

Template used: FMTID\_SummaryInformation -> PIDSI\_TEMPLATE

Revision number: FMTID\_SummaryInformation -> PIDSI\_REVNUMBER

Total editing time: FMTID\_SummaryInformation -> PIDSI\_EDITTIME

Last saved: FMTID\_SummaryInformation -> PIDSI\_LASTSAVE\_DTM

Last edited by: FMTID\_SummaryInformation -> PIDSI\_LASTAUTHOR

Last printed: FMTID\_SummaryInformation -> PIDSI\_LASTPRINTED

Recent hyperlinks list:

Page count:

Word count: FMTID\_SummaryInformation -> PIDSI\_WORDCOUNT

Character count: FMTID\_SummaryInformation -> PIDSI\_CHARCOUNT

Track changes:

Slide notes:

GUID:

### Retrieving content text

Microsoft provides their proprietary file formats in this link: <http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=14565>

The system took up methodologies to parse the data files: doc, ppt, xls. There are detailed descriptions to extract contents text.

* From DOC file

There is a detailed description how to retrieve text content form DOC binary in the documentation **[MS-DOC]: Word (.doc) Binary File Format** [2.4.1 Retrieving Text].

* From PPT file

PPT binary file consists of number of **Records** which has flags, type, length and variable data fields. There are various types of **container record**s which can involve other type of record. 2 Types of records are able to take text content: **RT\_TextByteAtom**, **RT\_TextCharsAtom**

Following algorithm specifies how to extract content text from the PPT binary file.

1. Loop for each records until last record of the file.
2. Read *flags*, *type*, *length* fields and seek to read bytes.
3. If *flags* specify that the record is container, seek to *length* bytes. If not, continue loop[1].
4. If *type* is RT\_TextByteAtom or RT\_TextCharsAtom, then read following records *data* fields as the *length* and store as string.

* From XLS file

XLS binary file consists of number of **Record** which has type, length and variable data fields. There are various types of **cell record**s which involves cell data: **LabelSST**, **Number**, **MulRk**, **Rk**.

**LabelSST** record has an index of String table included in **SST** record which stores shared strings all over the documentation.

Following algorithm specifies how to extract content text from the PPT binary file.

1. Loop for each records until last record of the file.
2. Read *type*, *length* fields and seek to read bytes.
3. If type specifies a SST record, read **shared strings** from data area and save them in *shared string array.*
4. If *type* specifies a LabelSST record, read index to shared string table from data area and get a string from extracted *shared string array*[3].
5. If *type* specifies a cell record (MulRk, Rk) read following records *data* fields as the *length* and store as string.
6. If *type* specifies a cell record (LabelSST) read following records *data* fields as the *length* and store as string.

## MS Office 2007

Office 2007 files are formatted by Open Office XML standard: The general format is Zip archive which involves number of xml files in it. The file [Content\_Types].xml has the paths to specific xml files including various data.

Followings are the values of attribute *ContentType* for a node to specify the file paths in the archive:

* Application specific properties

*application/vnd.openxmlformats-officedocument.extended-properties+xml*

* General properties

*application/vnd.openxmlformats-package.core-properties+xml*

* Word Document

*application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml*

* Powerpoint Slides

*application/vnd.openxmlformats-officedocument.presentationml.slide+xml*

* Excel Workseets

*application/vnd.openxmlformats-officedocument.spreadsheetml.worksheet+xml*

* Excel Shared Strings

*application/vnd.openxmlformats-officedocument.spreadsheetml.sharedStrings+xml*

### Retrieving meta info

* Following meta data can be extracted from **General properties** xml file:

Author, Revision, Last modified, Title…

* Following meta data can be extracted from **Application specific properties** xml file:

Company, Application, Template, Total time, Recent hyperlinks list, Page count, Word count, Character count

It is like follows considering each document type.

* From DOCX file

Author name: [General properties xml] -> dc:creator

Title: [General properties xml] -> dc:title

Subject: [General properties xml] -> dc:subject

Keywords: [General properties xml] -> cp:keywords

Status: [General properties xml] -> cp:contentStatus

Company: [Application specific properties xml] -> Company

Application: [General properties xml] -> Application

Comments: [General properties xml] -> dc:description

Template used: [General properties xml] -> Template

Revision number: [General properties xml] -> cp:revision

Total editing time: [Application specific properties xml] -> TotalTime

Last saved: [Application specific properties xml] -> dcterms:modified

Last edited by: [Application specific properties xml] -> cp:lastModifiedBy

Last printed: [Application specific properties xml] -> cp:lastPrinted

Recent hyperlinks list: [Application specific properties xml] -> HyperlinksChanged

Page count: [Application specific properties xml] -> Pages

Word count: [Application specific properties xml] -> Words

Character count: [Application specific properties xml] -> Characters

Track changes:

Slide notes:

GUID:

* From PPTX file

Author name: [Application specific properties xml] -> dc:creator

Title: [General properties xml] -> dc:title

Subject: [General properties xml] -> dc:subject

Keywords: [General properties xml] -> cp:keywords

Status: [General properties xml] -> cp:contentStatus

Company: [General properties xml] -> Company

Application: [General properties xml] -> Application

Comments: [General properties xml] -> dc:description

Template used: [General properties xml] -> Template

Revision number: [Application specific properties xml] -> cp:revision

Total editing time: [General properties xml] -> TotalTime

Last saved: [Application specific properties xml] -> dcterms:modified

Last edited by: [Application specific properties xml] -> cp:lastModifiedBy

Last printed: [Application specific properties xml] -> cp:lastPrinted

Recent hyperlinks list: [General properties xml] -> HyperlinksChanged

Page count:

Word count: [General properties xml] -> Words

Character count:

Track changes:

Slide notes: [Application specific properties xml] -> Slides

GUID:

* From XLSX file

Author name: [Application specific properties xml] -> dc:creator

Title: [General properties xml] -> dc:title

Subject: [General properties xml] -> dc:subject

Keywords: [General properties xml] -> cp:keywords

Status: [General properties xml] -> cp:contentStatus

Company: [General properties xml] -> Company

Application: [General properties xml] -> Application

Comments: [General properties xml] -> dc:description

Template used: [General properties xml] -> Template

Revision number: [Application specific properties xml] -> cp:revision

Total editing time: [General properties xml] -> TotalTime

Last saved: [Application specific properties xml] -> dcterms:modified

Last edited by: [Application specific properties xml] -> cp:lastModifiedBy

Last printed: [Application specific properties xml] -> cp:lastPrinted

Recent hyperlinks list: [General properties xml] -> HyperlinksChanged

Page count:

Word count:

Character count:

Track changes:

Slide notes:

GUID:

### Retrieving content text

* From DOC file

The text contents of DOC file are stored in *Word Document* xml file with following tags:

*w:p* paragraph

*w:tab* tab

*w:r*, *w:cr*, *w:br* new lines, breaks

*w:t* text

1. Open *Word Document* xml file.
2. Loop for every xml nodes in the opened file[1].
3. Read node’s data according to the node tags.

* From PPT file

The text contents of PPT file are stored in *Powerpoint Slides* xml files with following tags:

*a:p* paragraph

a*:tab* tab

a*:r*, a*:cr*, a*:br* new lines, breaks

a*:t* text

1. Open *Powerpoint Slides* xml file.
2. Loop for every xml nodes in the opened file[1].
3. Read node’s data according to the node tags.

* From XLS file

The text contents of XLS file are stored in *Excel Workseets* xml files with following tags:

*row* row

*c* cell

*v* value

1. Open *Excel Shared Strings* xml file
2. Extract shared strings from opened file[1] and store it in a array.
3. Open *Word Document* xml file.
4. Loop for every xml nodes in the opened file[3].
5. Read node’s data according to the node tags. If value type (tag *t*) is “s”, get string from shared string array[2] by the index value.

## MS RTF

Rich Text Format is specified by Microsoft and current version is 1.9.1 and published at:

<http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=10725>

In the specification documentation an Appendix A: Sample RTF Reader Application is included with detailed description about how to write an RTF Reader and a sample implementation source code.

This system is including sample RTF reader source code and customizing it with overriding a function *ecPrintChar* to extract content text.

## Text

In Text file, there is first 2~3 bytes optional header to specify the unicode encoding and endian. If the encoding is Ascii, there is no header in the file.

It is known as a BOM(Byte Oder Mark) such as the following constants:

* "\xef\xbb\xbf", 3bytes, L"UTF-8"
* "\xff\xfe", 2bytes, L"UTF-16LE"
* "\xfe\xff", 2bytes, L"UTF-16BE"

If the first 2~3 bytes doesn’t match any of the BOM table entries, it is assumed as a Ascii encoding text. Following byte stream are converted to UTF8 text under its encoding and endian specified by BOM.

## Adobe PDF

Adobe PDF file format is published at:

<http://www.adobe.com/devnet/pdf/pdf_reference.html>.

The system is including an open source library (SumatraPDF-1.8) to parse and extract basic pdf elements and including external library dependencies such as zlib, fitz.

### Retrieving meta info

PDF file consists of primarily of objects, of which there are eight types:

* [Boolean](http://en.wikipedia.org/wiki/Boolean_data_type) values, representing true or false
* Numbers
* [Strings](http://en.wikipedia.org/wiki/String_(computer_science))
* Names
* [Arrays](http://en.wikipedia.org/wiki/Array_data_type), ordered collections of objects
* Dictionaries, collections of objects indexed by Names
* [Streams](http://en.wikipedia.org/wiki/Stream_(computing)), usually containing large amounts of data
* The [null](http://en.wikipedia.org/wiki/Pointer_(computing)) object

In a PDF file, the meta information are ususally presented in a number of *Dictionary* objects distinguished by *key*. Here are name of major root *dictionaries* which can involves a kind of meta information and other dictionaries:

* *Root*

A catalog dictionary, main and most top level dictionary which can include dictionaries and various type of objects for meta and contents data.

The system extracts XMP, page tree, embedded files and attachments from this dictionary.

Information about XMP location can be found at the following pdf:

*Adobe, “Embedding XMP Metadata in Application Files”, September 14, 2001*

* *Info*

An information dictionary, most top level dictionary including objects for various meta information such as title, subject, author, keywords, creation date, modification date.

Here shows the object key hierarchy to locate the meta data in pdf file format:

|  |  |
| --- | --- |
| Target data | Object key hierachy (key names tree) |
| Title | [Info] -> [Title] |
| Subject | [Info] -> [Subject] |
| Author | [Info] -> [Author] |
| Keywords | [Info] -> [Keywords] |
| Creation date | [Info] -> [CreationDate] |
| Modification date | [Info] -> [ModDate] |
| XMP | [Root] -> [Metadata] |
| Page reference | [Root] -> [Pages] |
| Page count | [Root] -> [Pages] -> [Count] |
| Embedded files and attachments | [Root] -> [EmbeddedFiles] |
| Review and comments | [Root] -> [Pages] -> [Annots] |

### Retrieving content text

There is description about how to get page tree at a pdf file format. Once the page tree loaded, it is able to read each stream in each page object. There are several tokens and keywords which must be considered to extract text contents. First the stream data should be divided by tokens and inside a specific token content the data must be distinguished by keywords.

Following tokens must be considered to extract text contents from a pdf file.

* *(*

This token specifies that a string is starting from the following byte.

* *)*

This token specifies that a string has been ended.

Following keywords must be considered to extract text contents from a pdf file.

* *BT*

This keyword specifies that a text data is starting from the following byte.

* *ET*

This keyword specifies that a text data has been ended.

## JPEG, TIFF

Almost image files support in EXIF standard. The system extracts the following information from EXIF data. The detailed format specification can be found at:

<http://www.exif.org/specifications.html>

The system is including an open source library exiv2 which supports widely used image formats. Once exiv2 is called to extract EXIF data from every image file and the system divides the extracted EXIF data into individual meta data.

Followings are the EXIF keys corresponding to each meta data.

|  |  |
| --- | --- |
| Target data | EXIF keys |
| Photo taken time | Exif.Photo.DateTimeOriginal |
| Description | Exif.Photo.UserComment |
| Camera setting | Exif.Image.Make  Exif.Image.Model  Exif.Photo.FNumber  Exif.Photo.ExposureTime  Exif.Photo.ISOSpeedRatings  Exif.Photo.ExposureBiasValue  Exif.Photo.FocalLength  Exif.Photo.MaxApertureValue  Exif.Photo.MeteringMode  Exif.Photo.Flash |
| Geo tag | Exif.GPSInfo.GPSVersionID  Exif.GPSInfo.GPSLatitudeRef  Exif.GPSInfo.GPSLatitude  Exif.GPSInfo.GPSLongitudeRef  Exif.GPSInfo.GPSLongitude  Exif.GPSInfo.GPSTimeStamp  Exif.GPSInfo.GPSMapDatum |