

A decorative graphic on the right side of the page. It features three sets of concentric circles in shades of blue. The top set is the largest, the middle set is the smallest, and the bottom set is the largest. Two thin blue lines intersect at the center of the middle set of circles, extending towards the top-left and bottom-right corners of the page.

# Documentación Externa

## III Tarea Programada

Lenguajes de Programación

Yader Morales López  
Ariel Mora Matamoros  
Joshua Hernández Bonilla

**29/05/2012**

## Tabla de contenidos

Tabla de contenidos .....	2
Descripción del problema .....	3
Diseño del programa .....	4
Decisiones Tomadas.....	5
Estructuras de Datos .....	6
Lógica del Programa .....	7
Análisis de resultados.....	9
Librerías usadas.....	9
Manual de usuario (instrucciones uso, de ejecución y de compilación).....	10
Conclusión personal .....	16
Bibliografía .....	17

## Descripción del problema

Se nos plantea la creación de un programa que sea capaz de leer metadatos de archivos PDF, con el fin de poder dar al usuario facilidades de búsqueda en este tipo de documentos. La mayoría de las ocasiones se mantienen muchos de estos documentos juntos y por sólo el nombre del archivo no se conoce el contenido, lo que lleva a abrir el documento y leer los datos del mismo.

Este programa se encargará de leer el título, fecha de creación, palabras claves y el autor, el cual será desplegado en consola, además deberá de poder permitir al usuario dejar insertar palabras en la búsqueda para realizar un filtro en los documentos. Esta lectura de datos, deberá ser almacenada en una estructura de Lisp, para que pueda ser consultada. El programa deberá además de mostrar al usuario los siguientes datos:

- Nombre del archivo
- Título
- Asunto
- Autor
- Fecha de creación
- Palabras clave

Estos puntos deben ser mostrados en formato tabla al usuario y deberán de recibir algún parámetro de búsqueda, si no se les da, debe mostrar todos los archivos de acuerdo al modo de consulta.

## Diseño del programa

Este programa consiste en la elaboración de una aplicación basada en el lenguaje de programación Lisp, la cual radica en la elaboración de una base de datos de información de archivos PDF. Esta aplicación primero extrae de una carpeta indicada por el usuario la meta información de los headers de los archivos PDF, la decodifica en información entendible, para que luego sea almacenada en objetos de tipo PDF, con sus respectivos atributos, como Título, Autor, Fecha de Creación y Palabra clave.

Luego esos objetos son agregados a una tabla hash con una correspondiente llave para poder realizar consultas por Título, Autor, Fecha de creación y palabras clave, para mostrar al usuario las coincidencias encontradas.

Las funciones utilizadas en la aplicación son las siguientes:

- `component-present-p`: La cual verifica si se encuentra el valor que se envía por parámetro.
- `directory-pathname-p`: Utiliza `component-present-p` para verificar las rutas de los archivos.
- `pathname-as-directory`: Se le envía una `pathname` y lo transforma en un directorio, por si existen carpetas dentro de la dirección indicada.
- `list-directory`: Verifica las direcciones que se le envía por parámetros.
- `directory-wildcard`: Se le envía una dirección y se encarga de crear un `pathname`.
- `walk-directory`: Función que recibe un directorio por el cual va a recorrer los tipos de archivos que se le indiquen, para esta aplicación busca todos los archivos con extensión `.pdf`.
- `pdf-p`: Se encarga de verificar si un archivo es de extensión `.pdf`.
- `recorrerFichero`: Recibe el nombre del archivo y se encarga de llamar a una función con la cantidad de bytes a leer.
- `recorrerFichero-aux`: Recibe el nombre del archivo y la longitud a leer y se encarga de abrir el fichero y mediante la ayuda de la función `buscarTag`, busca que tipo de tag es, si es un título, autor, fecha de creación o keyword y mediante el uso de `subseq` se extrae el texto perteneciente a ese tag.
- `buscarTag`: utiliza la función `search` de Lisp que retorna la posición en la cual se encuentra un texto dentro de otro.
- `buscaCod`: Recibe dos tags y un String que buscar para buscar el siguiente más contínuo para poder cortar mediante la ayuda de la función de Lisp `subseq` y las posiciones de los tags en el string.
- `buscar-en`: Recibe una dirección, en la que va a utilizar la función `walk-directory` para recorrer todos los archivos de tipo PDF, esta es la primera función que llama el usuario para cargar una carpeta llena de archivos con extensión PDF.
- `buscar-titulo`: recibe un título a buscar y devuelve todas las coincidencias que encuentre respecto a ese título. Utiliza la función `imprime-titulo` la cual recibe el PDF y el título a mostrar y lo muestra en consola.

- buscar-creador: recibe un autor a buscar y devuelve todas las coincidencias que encuentre respecto a ese autor. Utiliza la función imprime-creador la cual recibe el PDF y el autor a mostrar y lo muestra en consola.
- buscar-fechacreacion: recibe una fecha a buscar y devuelve todas las coincidencias que encuentre respecto a esa fecha. Utiliza la función imprime-fecha la cual recibe el PDF y la fecha a mostrar y la muestra en consola.
- buscar-keyword: recibe un keyword a buscar y devuelve todas las coincidencias que encuentre respecto a ese keyword. Utiliza la función imprime-keyword la cual recibe el PDF y el keyword a mostrar y lo muestra en consola.

## Decisiones Tomadas

1. Dentro de las decisiones tomadas dentro del grupo de trabajo fue establecer Ubuntu como sistema operativo y utilizar el Common Lisp para este Sistema operativo.
2. Utilizar el libro Practical Common Lisp como principal referencia de código, este libro es proporcionado por la pagina <http://www.gigamonkeys.com/book/>, y con base a los capítulos 15, 24 y 25 realizamos la tarea programada.
3. De los tags de los PDF, únicamente tomamos el título, autor, fecha de creación y keywords, ya que son los atributos necesarios para las búsquedas.
4. Decidimos utilizar varias funciones provistas por el Practical Common Lisp, ya que consideramos que es una buena manera de implementarlo y además somos capaces de entenderlas y las adaptamos para nuestro beneficio.
5. Utilizamos las clases vistas en clase, así como las funciones que se implementaron para recorrer las tablas hash.

## Estructuras de Datos

Se diseñó la clase file-pdf, esta clase contiene el nombre del archivo, autor, fecha de creación y el keyword. Para guardar cada PDF mínimo debe de tener de un título para crear el objeto, luego de crear el objeto PDF, ese objeto puede utilizar métodos de set and get para obtener cada atributo de ese objeto, solo se necesita enviar el objeto y llamar a la función correspondiente para obtener la información deseada.

Se utiliza el método que tiene Lisp de Open para leer un archivo y luego se utiliza code-char para obtener los chars correspondientes a esos números, con esto se pueden extraer los tags que van a almacenarse en los objetos y luego se ingresan a la tabla hash para recorrerla.

Todo el manejo de los tags se maneja mediante objetos y atributos, estos objetos se introducen dentro de una tabla hash indicando una llave para acceder a cada objeto. Este programa se encuentra estructurado según las consultas que se deben realizar sobre nombre del archivo, autor, fecha de creación y keyword, primero se cargan los datos para que el sistema tenga las tablas hash cargadas según los archivos que se encuentran en un directorio. Luego a partir de funciones predefinidas en Lisp como maphash para recorrer y sacar cada PDF para comprobar cada consulta que se realiza y mostrarle al usuario los PDF's que cumplen con las características.

## Lógica del Programa

La lógica del programa consiste en extraer los metadatos de un archivo PDF, esta información se encuentra en formato binario por lo cual se debió hacer funciones que decodificaran la información presente en los header de los archivos PDF mediante el uso de code-char,

```
(dotimes (i (file-length in))  
  
  (setf (char string i) (code-char (read-byte in))))
```

Estas funciones se realizaron utilizando la lógica de buscar las etiquetas que aparecen a la información decodificada, ya que existe un formato para la describir cada etiqueta, con lo cual se puede saber qué tipo de información se está leyendo. Según el libro en el que nos basamos, Practical Common Lisp, se podía extraer utilizando frames pero no logramos implementar esta manera de obtener la información, sino más bien utilizamos en la misma función los nombres de los tags a buscar, y mediante el método search buscamos si algún tag aparecía y además verificábamos si la información del tag era de utilidad.

```
(dotimes (i (file-length in))  
(setf (char string i) (code-char (read-byte in))))  
(setf (gethash *contador* *ht*) (make-instance 'file-pdf  
  :file-name (if (NULL (buscarTag "Title" string)) "Ninguno"  
    (buscaCod "Title" "/Author" string))  
  :author-name (if (NULL (buscarTag "Author" string)) "Ninguno"  
    (buscaCod "Author" "/Creator" string))  
  :keywords-name (if (NULL (buscarTag "Keywords" string)) "Ninguno"  
    (buscaCod "Keywords" "/APPL:Keywords" string))  
  :CreateDate-name (if (NULL (buscarTag "CreationDate" string)) "Ninguno"  
    (buscaCod "CreationDate" "/ModDate" string))))
```

Mediante ese código elaborábamos un string con cada información de los tags.

Luego de realizar esta extracción de datos se carga la información en objetos de tipo PDF, según las clases vistas en clase, luego esos objetos se cargan a memoria, para esto se utiliza tablas hash las cuales mediante una llave se puede acceder a la información valor, en nuestro caso, los PDF con la información, y cada llave consisten en un valor numérico consecutivo.

Para el manejo de las diferentes consultas que se deben de realizar, por nombre de archivo, autor, fecha de creación y keyword, se emplea la función maphash, que me permite extraer la información de la tabla hash, para luego mediante el método gethash nombre de la tabla y lo que

se desea extraer se compara con lo que el usuario escribió, y cuando se encuentren coincidencias se muestran al usuario, esto para cada tipo de búsqueda que desee realizar el usuario.

Mediante esa función se imprimían todas las canciones de la base de datos, para ello se utiliza la función maphash y un lambda para obtener solo los valores de esa tabla hash. Luego mediante `imprime-titulo/credor/fecha/keyword` se mostraba al usuario en consola.

```
(defun buscar-titulo(titulo) ;imprime todos las canciones con un titulo en comun
  (maphash #'(lambda (k v) (imprimir-titulo v titulo)) *ht*))

(defun imprimir-titulo (file titulo)
  (if (search titulo (getTitleName file)) (format t
    "Autor: ~a Palabras clave: ~a Fecha de creacion: ~a ~%"
    (getAuthorName file)(getKeyword file)(getDateName file))))
;;;;
```

Y funciona de igual manera para la consulta de autor, fecha de creación y keyword.



## **Análisis de resultados**

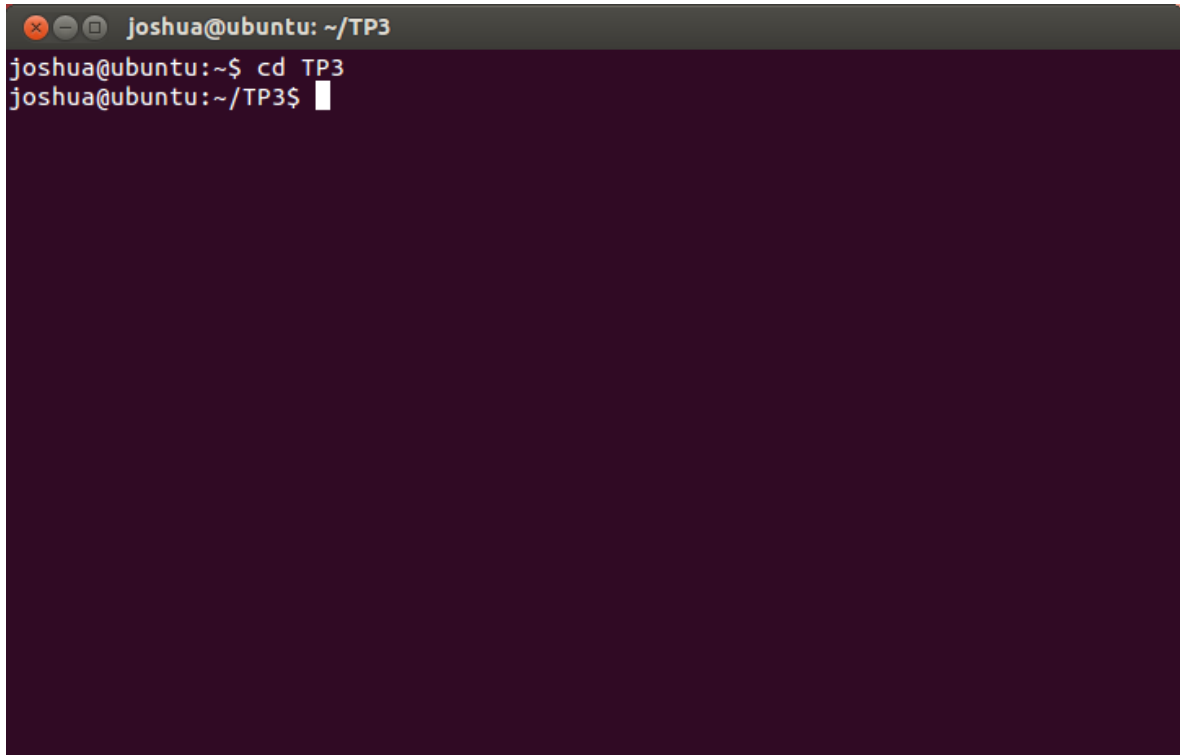
- El programa funciona sólo con el estándar PDF 1.5, ya que los métodos para obtener los metadatos de los otros estándares es diferente.
- Nuestro programa es capaz de crear una base de datos mediante un hashtable de los PDF's ingresados. Se debe tener en cuenta que la tabla hash fue creada mediante objetos.
- El programa funciona sólo con archivos PDF sin importar la extensión del mismo siempre y cuando los PDF cumplan con el estándar 1.5.
- El programa realiza consultas de los autores, títulos, fechas de creación y keywords de los PDF's localizados en una carpeta.
- Cada vez que se cierra la aplicación, se pierde la información ingresada a la base de datos.
- Al cerrar el programa, el mismo no se cicla ni quedan procesos abiertos.

## **Librerías usadas**

En esta tarea programada no se utilizan librerías.

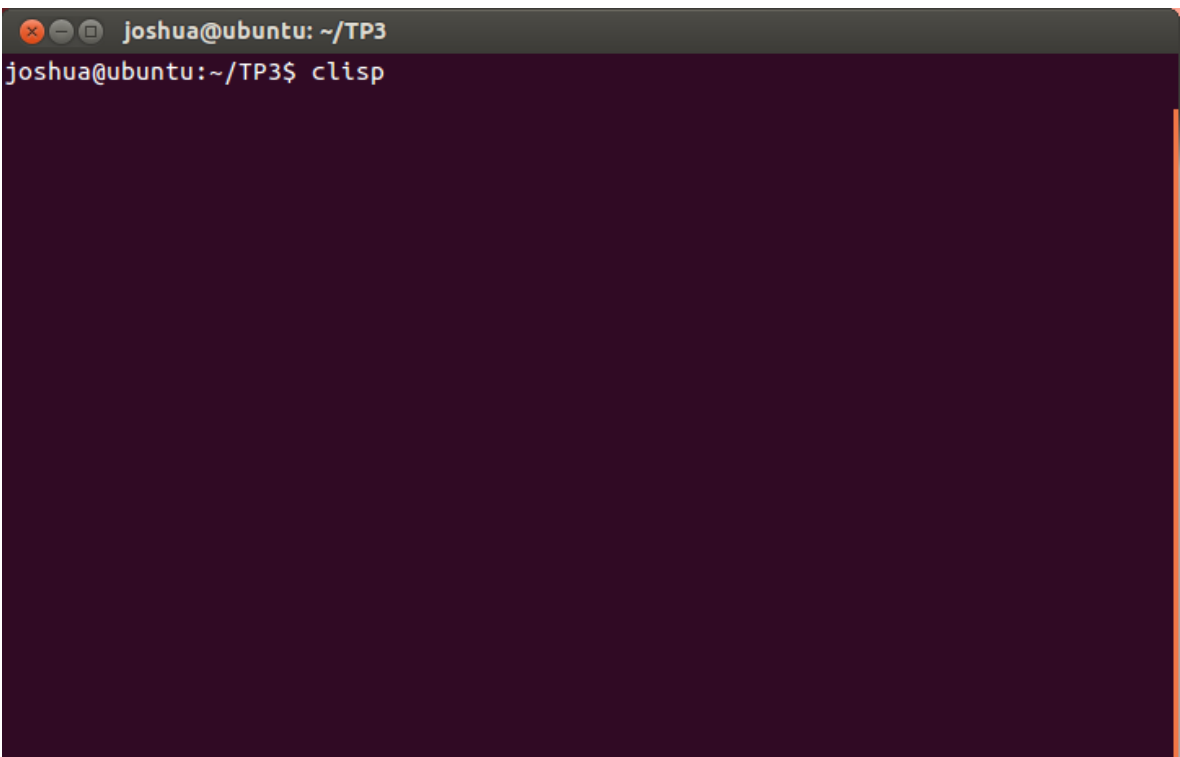
## Manual de usuario (instrucciones uso, de ejecución y de compilación)

1. Primero accedemos a la carpeta en donde se encuentra el archivo cd "nombre de carpeta", como se muestra a continuación:

A terminal window with a dark purple background. The title bar shows 'joshua@ubuntu: ~/TP3'. The command history shows 'joshua@ubuntu:~\$ cd TP3' followed by 'joshua@ubuntu:~/TP3\$' with a cursor.

```
joshua@ubuntu: ~/TP3
joshua@ubuntu:~$ cd TP3
joshua@ubuntu:~/TP3$
```

2. Cargar el interprete de Lisp (Clisp)

A terminal window with a dark purple background. The title bar shows 'joshua@ubuntu: ~/TP3'. The command history shows 'joshua@ubuntu:~/TP3\$ clisp' followed by a blank line.

```
joshua@ubuntu: ~/TP3
joshua@ubuntu:~/TP3$ clisp
```

3. Compilar el programa con (compile-file "Nombre de Archivo"). Como se muestra a continuación:

```
joshua@ubuntu: ~/TP3
joshua@ubuntu:~/TP3$ clisp
  i i i i i i
  I I I I I I   8 8 8 8 8 8 8 8 8 8
  I \ \ '+' / I   8 8 8 8 8 8 8 8
  \ \ -+-' /      8 8 8 8 8 8 8 8
  - _ | _ -'      8 8 8 8 8 8 8 8
  |      8 8 8 8 8 8 8 8
  -----+----- 00000 8000000 0008000 00000 8

Welcome to GNU CLISP 2.49 (2010-07-07) <http://clisp.cons.org/>

Copyright (c) Bruno Haible, Michael Stoll 1992, 1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2000
Copyright (c) Sam Steingold, Bruno Haible 2001-2010

Type :h and hit Enter for context help.

[1]> (compile-file "TP3.lisp")
```

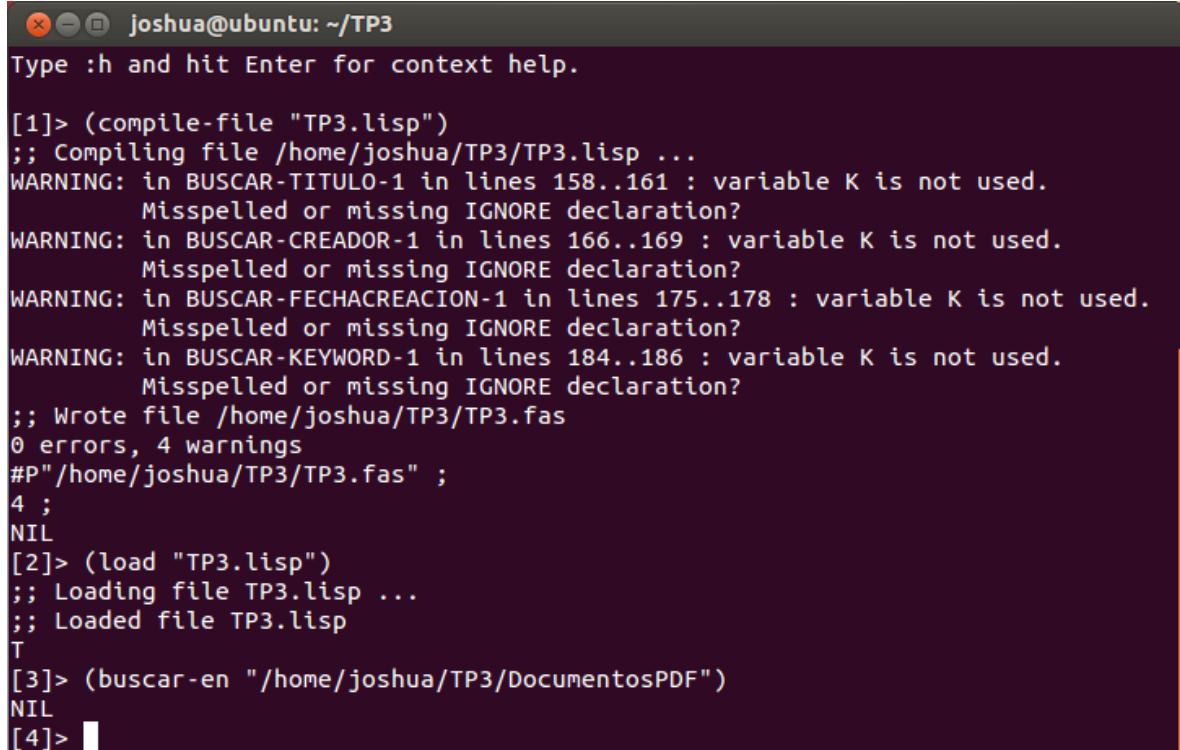
4. Cargar el programa con (load "Nombre de Archivo"). Como se muestra a continuación:

```
joshua@ubuntu: ~/TP3
Copyright (c) Bruno Haible, Michael Stoll 1992, 1993
Copyright (c) Bruno Haible, Marcus Daniels 1994-1997
Copyright (c) Bruno Haible, Pierpaolo Bernardi, Sam Steingold 1998
Copyright (c) Bruno Haible, Sam Steingold 1999-2000
Copyright (c) Sam Steingold, Bruno Haible 2001-2010

Type :h and hit Enter for context help.

[1]> (compile-file "TP3.lisp")
;; Compiling file /home/joshua/TP3/TP3.lisp ...
WARNING: in BUSCAR-TITULO-1 in lines 134..137 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-CREADOR-1 in lines 142..145 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-FECHACREACION-1 in lines 151..154 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-KEYWORD-1 in lines 160..162 : variable K is not used.
        Misspelled or missing IGNORE declaration?
;; Wrote file /home/joshua/TP3/TP3.fas
0 errors, 4 warnings
#P"/home/joshua/TP3/TP3.fas" ;
4 ;
NIL
[2]> (load "TP3.lisp")
```

5. Ingresar el la dirección en que se desea buscar PDF's, con (buscar-en "Dirección"). Como se muestra a continuación:



```
joshua@ubuntu: ~/TP3
Type :h and hit Enter for context help.

[1]> (compile-file "TP3.lisp")
;; Compiling file /home/joshua/TP3/TP3.lisp ...
WARNING: in BUSCAR-TITULO-1 in lines 158..161 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-CREADOR-1 in lines 166..169 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-FECHACREACION-1 in lines 175..178 : variable K is not used.
        Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-KEYWORD-1 in lines 184..186 : variable K is not used.
        Misspelled or missing IGNORE declaration?
;; Wrote file /home/joshua/TP3/TP3.fas
0 errors, 4 warnings
#P"/home/joshua/TP3/TP3.fas" ;
4 ;
NIL
[2]> (load "TP3.lisp")
;; Loading file TP3.lisp ...
;; Loaded file TP3.lisp
T
[3]> (buscar-en "/home/joshua/TP3/DocumentosPDF")
NIL
[4]> 
```

Cuando se ingresa la dirección por defecto se retorna un NIL.

6. Realizar la consulta que se requiere mediante (buscar-titulo), (buscar-creador), (buscar-fechacreacion), (buscar-keyword).

La búsqueda por título se muestra a continuación:

```
joshua@ubuntu: ~/TP3
;; Compiling file /home/joshua/TP3/TP3.lisp ...
WARNING: in BUSCAR-TITULO-1 in lines 158..161 : variable K is not used.
Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-CREADOR-1 in lines 166..169 : variable K is not used.
Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-FECHACREACION-1 in lines 175..178 : variable K is not used.
Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-KEYWORD-1 in lines 184..186 : variable K is not used.
Misspelled or missing IGNORE declaration?
;; Wrote file /home/joshua/TP3/TP3.fas
0 errors, 4 warnings
#P"/home/joshua/TP3/TP3.fas" ;
4 ;
NIL
[2]> (load "TP3.lisp")
;; Loading file TP3.lisp ...
;; Loaded file TP3.lisp
T
[3]> (buscar-en "/home/joshua/TP3/DocumentosPDF")
NIL
[4]> (buscar-titulo "Inteligencia Artificial")
Autor: ðJoshua Hernández Bonilla) Palabras clave: Ninguno Fecha de creacion: D
:20120602204206-06'00')
NIL
[5]>
```

La búsqueda por autor es la siguiente:

```
joshua@ubuntu: ~/TP3
Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-FECHACREACION-1 in lines 175..178 : variable K is not used.
Misspelled or missing IGNORE declaration?
WARNING: in BUSCAR-KEYWORD-1 in lines 184..186 : variable K is not used.
Misspelled or missing IGNORE declaration?
;; Wrote file /home/joshua/TP3/TP3.fas
0 errors, 4 warnings
#P"/home/joshua/TP3/TP3.fas" ;
4 ;
NIL
[2]> (load "TP3.lisp")
;; Loading file TP3.lisp ...
;; Loaded file TP3.lisp
T
[3]> (buscar-en "/home/joshua/TP3/DocumentosPDF")
NIL
[4]> (buscar-titulo "Inteligencia Artificial")
Autor: ðJoshua Hernández Bonilla) Palabras clave: Ninguno Fecha de creacion: D:20
120602204206-06'00')
NIL
[5]> (buscar-creador "Jose")
Nombre de archivo: ðSistema de Gestión de Riesgos SIGERI) Palabras clave: Ninguno
Fecha de creacion: D:20120602213237-06'00')
NIL
[6]>
```

La búsqueda por fecha es la siguiente:

```
joshua@ubuntu: ~/TP3
Misspelled or missing IGNORE declaration?
;; Wrote file /home/joshua/TP3/TP3.fas
0 errors, 4 warnings
#P"/home/joshua/TP3/TP3.fas" ;
4 ;
NIL
[2]> (load "TP3.lisp")
;; Loading file TP3.lisp ...
;; Loaded file TP3.lisp
T
[3]> (buscar-en "/home/joshua/TP3/DocumentosPDF")
NIL
[4]> (buscar-titulo "Inteligencia Artificial")
Autor: pyJoshua Hernández Bonilla) Palabras clave: Ninguno Fecha de creacion: D:20120602204206-06'00')
NIL
[5]> (buscar-creador "Jose")
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Palabras clave: Ninguno
Fecha de creacion: D:20120602213237-06'00')
NIL
[6]> (buscar-fechacreacion "D:20120602213237-06'00'")
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Autor: Jose) /Subject(pyDo
cumentación de Análisis y Diseño) Palabras clave: Ninguno
NIL
[7]>
```

La búsqueda por keyword se muestra a continuación:

```
joshua@ubuntu: ~/TP3
NIL
[5]> (buscar-creador "Jose")
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Palabras clave: Ninguno
Fecha de creacion: D:20120602213237-06'00')
NIL
[6]> (buscar-fechacreacion "D:20120602213237-06'00'")
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Autor: Jose) /Subject(pyDo
cumentación de Análisis y Diseño) Palabras clave: Ninguno
NIL
[7]> (buscar-keyword "")
Nombre de archivo: "pySistema de Gestión de Riesgos SIGERI) " Autor: Jose) /Subject(pyDo
cumentación de Análisis y Diseño) Fecha de creacion: D:20120602213237-06'00')
Nombre de archivo: "Inteligencia Artificial) " Autor: pyJoshua Hernández Bonilla) Fecha
de creacion: D:20120602204206-06'00')
Nombre de archivo: "Ninguno" Autor: biblioguest) Fecha de creacion: D:20120602213505-06
'00')
Nombre de archivo: "pyLa Globalización) " Autor: pyJoshua Hernández Bonilla) Fecha de c
reacion: D:20120602204140-06'00')
Nombre de archivo: "Ninguno" Autor: almora) Fecha de creacion: D:20120602213601-06'00')
Nombre de archivo: "Ninguno" Autor: jstradi) Fecha de creacion: D:20120602203753-06'00'
)
NIL
[8]>
```

7. Si se desea salir del programa ingresar (exit). Como se muestra a continuación:

```
joshua@ubuntu: ~/TP3
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Palabras clave: Ninguno
Fecha de creacion: D:20120602213237-06'00')
NIL
[6]> (buscar-fechacreacion "D:20120602213237-06'00'")
Nombre de archivo: pySistema de Gestión de Riesgos SIGERI) Autor: Jose) /Subject(pyDo
cumentación de Análisis y Diseño) Palabras clave: Ninguno
NIL
[7]> (buscar-keyword "")
Nombre de archivo: "pySistema de Gestión de Riesgos SIGERI) " Autor: Jose) /Subject(pyDo
cumentación de Análisis y Diseño) Fecha de creacion: D:20120602213237-06'00')
Nombre de archivo: "Inteligencia Artificial) " Autor: pyJoshua Hernández Bonilla) Fecha
de creacion: D:20120602204206-06'00')
Nombre de archivo: "Ninguno" Autor: biblioguest) Fecha de creacion: D:20120602213505-06
'00')
Nombre de archivo: "pyLa Globalización) " Autor: pyJoshua Hernández Bonilla) Fecha de c
reacion: D:20120602204140-06'00')
Nombre de archivo: "Ninguno" Autor: almora) Fecha de creacion: D:20120602213601-06'00')

Nombre de archivo: "Ninguno" Autor: jstradi) Fecha de creacion: D:20120602203753-06'00'
)
NIL
[8]> (exit)
Bye.
joshua@ubuntu:~/TP3$
```

## Conclusión personal

El sistema desarrollado logra cumplir con los objetivos planteados, además del entendimiento de un nuevo paradigma de programación. El manejo de información proveniente de un PDF es complicado, pues no todos los PDF se rigen bajo un mismo estándar donde se encontrarán los metadatos. Por ende se tuvo que adaptar al estándar 1.5 de PDF, el cual maneja los metadatos que nos interesan en un sitio específico del documento.

La utilización de Lisp como lenguaje de programación en esta tarea programada, abrió los conocimientos hacia un nuevo mundo de listas y notación prefija de cual se tenía poca experiencia, lo que añade un grado de complejidad mayor al programa planteado.

A pesar de los inconvenientes presentados en el desarrollo de la tarea programa, se logran los objetivos planteados y se desarrollan nuevos conocimiento del paradigma funcional.



## Bibliografía

- Seibel, P. (2009). GigaMonkeys. Recuperado el 23 de 05 de 2011, de <http://www.gigamonkeys.com/book/> Capítulos 14,24,25
- CLISP - Wikipedia, la enciclopedia libre (2012, 20 de Abril). Recuperado el 26 de Mayo del 2012, de <http://es.wikipedia.org/wiki/CLISP>
- Harvey, P. (2011, 01 de Diciembre). PDF Tags Recuperado el 26 de Mayo del 2012, de <http://www.sno.phy.queensu.ca/~phil/exiftool/TagNames/PDF.html>
- Cadenas - Guía rápida de Common Lisp (s. f.). Recuperado el 02 de Junio del 2012, de <http://lisp.manticore.es/manuales/guia-rapida-cl/Cadenas.html>
- Dalton, J. (s. f.). A Brief Guide to CLOS Recuperado el 02 de Junio del 2012, de <http://www.aiai.ed.ac.uk/~jeff/clos-guide.html>
- The Common Lisp Cookbook - Strings (2007, 28 de Enero). Recuperado el 02 de Junio del 2012, de <http://cl-cookbook.sourceforge.net/strings.html>
- Cadenas - Guía rápida de Common Lisp (s. f.). Recuperado el 02 de Junio del 2012, de <http://lisp.manticore.es/manuales/guia-rapida-cl/Cadenas.html>
- Lamkins, D. (1995). Successful Lisp - Chapter 19 Recuperado el 03 de Junio del 2012, de <http://www.psg.com/~dlamkins/sl/chapter19.html>
- LispWorks Ltd (1996). LispWorks Recuperado el 03 de Junio del 2012, de [http://www.lispworks.com/documentation/lw60/CLHS/Body/f\\_rd\\_by.htm](http://www.lispworks.com/documentation/lw60/CLHS/Body/f_rd_by.htm)
- Reading a file in common lisp - Stack Overflow (2010, 01 de Noviembre). Recuperado el 03 de Junio del 2012, de <http://stackoverflow.com/questions/4066516/reading-a-file-in-common-lisp>
- Togores Fernández, R. (1999). 2.2.6. Funciones Visual LISP ara Tratamiento de Cadenas - De Formas y Funciones Recuperado el 03 de Junio del 2012, de [http://www.togores.net/vl/curso/lisp/bases/funciones/cadenas\\_vl](http://www.togores.net/vl/curso/lisp/bases/funciones/cadenas_vl)