UNIVERSIDAD AUTONOMA DE MADRID

Escuela Politécnica Superior

20|21

Trabajo fin de grado

Análisis automatizado de metadatos expuestos en ficheros públicos

Antonio Solana Vera

Escuela Politécnica Superior
Universidad Autónoma de Madrid
C\Francisco Tomás y Valiente nº 11

excelencia Campus Internacional UAM CSIC

www.uam.es

UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Análisis automatizado de metadatos expuestos en ficheros públicos

Autor: Antonio Solana Vera
Tutor: Jaime Lopez Sánchez

abril 2021

Antonio Solana Vera
Análisis automatizado de metadatos expuestos en ficheros públicos

Antonio Solana Vera
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A mi familia y a mis amigos

Resumen

Los metadatos son una parte muy importante en nuestro día a día. A pesar de esto, no los usamos directamente. Solo otros programas se aprovechan de los metadatos de ciertos archivos para facilitar su uso y visualización. Es por esta razón que muchos de los archivos que se suben a internet contienen metadatos que pueden filtrar información sensible. Aunque las gran mayoría de las redes sociales y otros servicios de subida de archivos eliminan los metadatos de forma automática, muchas empresas más pequeñas no lo hacen. Y aunque estas empresas intentaran visualizar estos metadatos, actualmente no existe una herramienta o librería estándar que permita la recopilación automática de metadatos en páginas web de forma fácil.

A lo largo de este trabajo, se desarrollará un framework que sirva para la extracción de estos metadatos sin necesidad de tener conocimientos técnicos. Este framework, denominado "Krptkn", va a ser desarrollado en el lenguaje de programación Elixir con algunos módulos escritos en C para aprovechar algunas librerías interesantes.

Palabras clave

metadatos, seguridad, framework, tesis, trabajo fin de grado

Abstract

Metadata are a crucial part of our day to day. Despite this, we have no direct contact with them. Only other programs take advantage of the metadata of files to facilitate their use and offer us interesting information. Because of this, many of the files uploaded to the internet contain metadata that may be leaking sensitive information. Even though most of the commonly used social media networks do remove the metadata from your files automatically, many smaller or non-tech companies do not. And even if these smaller companies had an interest in checking the metadata they upload, there is no standard tool they can rely on.

In this project, a framework is going to be developed to help extract, filter and visualize this metadata without requiring technical knowledge. This framework is going to be developed in Elixir with some modules written in C to take advantage of existing libraries.

Keywords

metadata, security, framework, thesis, final degree project

ÍNDICE

Listas

Lista de algoritmos

Lista de códigos

Lista de cuadros

Lista de ecuaciones

Lista de figuras

Lista de tablas

Lista de cuadros

1

Introducción

Los metadatos son una clase de datos que nos dan información acerca de otros datos. Son secundarios al contenido de un archivo y normalmente su uso está dirigido a otro software y no a humanos. Algunos ejemplos de metadatos con los que estamos más familiarizados pueden ser las miniaturas de las fotos o los *tags* que nos permiten crear filtros de búsqueda.

Aunque la mayoría de metadatos son inofensivos en circunstancias normales, cuando existe un adversario que tiene intención de atacar nuestro sistema informático, los metadatos pueden darles una ventaja inusual. Por ejemplo: metadatos en nuestra página web que revelan que nuestro diseñador gráfico utiliza una versión vulnerable de Adobe Photoshop.

El objetivo de este proyecto es el desarrollo de una herramienta que facilite el descubrimiento de metadatos antes de que estos puedan suponer un problema de seguridad.

1.1. Fases de realización del proyecto

El desarrollo del proyecto consta de las siguientes fases:

- Planteamiento del problema: Formalización de los requisitos. Durante esta fase también se decide cuáles de los requisitos son necesarios y cuáles son opcionales.
- Estudio de soluciones existentes: Investigación de herramientas que solucionan el problema (completa o parcialmente).
- Diseño de la solución: Durante esta fase se decide el stack que se va a usar para el desarrollo de la herramienta. Se tienen en cuenta los requisitos decididos durante el planteamiento para elegir lenguaje, base de datos, plataforma, etc.
- Desarrollo: Desarrollo de la solución propuesta. Esta fase incluye también pruebas y revisiones.

A lo largo del proyecto ha habido algunos saltos entre fases, especialmente entre la fase de diseño y la de desarrollo, pero en general las decisiones con respecto a requisitos y diseño importantes se han mantenido estables.

1.2. Estructura del documento

La estructura de esta memoria se ha intentado que siga de forma cercana el mismo orden que las fases de desarrollo de la solución. Cada una de las fases vistas en 1.1 se corresponde con un capitulo de esta memoria de la siguiente manera:

- Planteamiento del problema - Capítulo 1 (Introducción).
- Estudio de soluciones existentes - Capítulo 2 (Estado del arte).
- Diseño de la solución - Capítulo 3 (Análisis de requisitos).
- Desarrollo - Capítulo 4 (Desarrollo de la solución).

Además la memoria consta de dos capítulos más: Resultados y Conclusiones y trabajo futuro. En estos exploraremos la solución final, sus defectos y posibles mejoras que se pueden aplicar.

2

Estado del arte

2.1. Introducción

En esta sección se van a analizar tres herramientas distintas que tienen en común el análisis de metadatos. Veremos las características principales que las hacen destacar y también lo que las hacen inservibles para nuestro caso de uso.

2.2. Foca

FOCA o Fingerprinting Organizations with Collected Archives cumple una tarea muy similar a la de Krptkn. Descarga archivos, extrae los metadatos (haciendo uso de una librería interna en lugar de una externa como Krptkn) y los presenta al usuario.

La principal diferencia entre Krptkn y FOCA es que FOCA utiliza los resultados de buscadores web como Google o DuckDuckGo para encontrar los archivos que analiza mientras que Krptkn tiene un crawler interno que se encarga de analizar la web objetivo en profundidad.

FOCA utiliza múltiples métodos para analizar un objetivo a parte de el uso de motores de búsqueda:

- Búsqueda a través de DNS. Se encuentran nuevos dominios e IPs a través de peticiones a servidores DNS. También se hacen uso de registros PTR en caso de que tengamos IPs que queramos escanear.
- Búsqueda recursiva. Cada IP nueva que se encuentra, se añade como nuevo objetivo y se analiza.
- Fuerza bruta. Se usan diccionarios para probar nombres comunes contra el servidor DNS.
- Predicción de DNS. Se detectan patrones de nombres para probar nuevas queries contra el servidor DNS.
- Robtex. Un servicio externo que se usa para encontrar nuevos dominios asociados con el objetivo.

A pesar de tener unas características tan atractivas, FOCA solo puede ser ejecutado en Windows y la base de datos que utiliza es SQL Server 2014 por lo que queda relegado a un uso más espontáneo. La herramienta es muy útil para redteams buscando hacer un análisis en profundidad de una empresa, pero no tanto para usarla como un servicio continuo.

2.3. National Library of New Zealand's Metadata Extraction Tool

La "Metadata Extraction Tool"de la biblioteca nacional de Nueva Zelanda tiene como objetivo la extracción de metadatos de archivos. No tiene funciones de análisis web ni crawling.

El proposito principal de esta herramienta era para uso interno en la biblioteca nacional, extrayendo metadatos de los archivos allí almacenados y preservarlos. Es por ello que no actúa como competidor directo a Krptkn pero sigue siendo importante analizarla para la parte de extracción, normalización y análisis de metadatos.

Las características más notorias de esta herramienta son las siguientes:

- Funcionamiento tanto en Windows como en Linux.
- Extracción de metadatos rápida. La herramienta analiza solo una pequeña parte de los archivos dados.
- Normalización de los metadatos a XML. Todos los metadatos extraídos son convertidos a XML para facilitar el almacenamiento y los procesos de búsqueda posteriores.
- Soporte para muchos formatos. El principal atractivo es la cantidad de archivos de oficina que puede analizar: MS Word, Word Perfect, Open Office, MS Works, MS Excel, MS PowerPoint y PDF.
- Cuenta con una interfaz de usuario escrita en Java y puede ser utilizada en forma de librería. Esto es ideal para nuestro caso de uso: una interfaz para visualizar resultados y una librería para un análisis no supervisado.

El principal inconveniente de esta herramienta es su antigüedad. La última versión es de 2014 y muchos de los formatos soportados no han sido probados con nuevas versiones.

2.4. Harvard's File Information Tool Set

FITS o File Information Tool Set, es una herramienta desarrollada por Harvard que agrega una docena de librerías todas dedicadas a la extracción de metadatos.

Algunas características de FITS:

- Gran cantidad de archivos soportados. Ya que agrega muchas herramientas distintas, la suma de todos los archivos soportados está en los miles.
- Tanto FITS como la mayoría de librerías que utiliza están escritas en Java y usan XML para normalizar los datos extraídos.
- La herramienta se actualiza una o dos veces al año. En general las actualizaciones son para usar las nuevas versiones de las librerías o por cambios de versión de Java.
- Puede ser utilizada en Windows, Mac o Linux.
- Tiene una API y una interfaz de terminal.

En general la herramienta es útil y proporciona una gran cantidad de metadatos sobre muchos

formatos distintos. La falta de una librería podría ser ignorada en favor de un despliegue conjunto de Krptkn y FITS, dejando la parte de análisis web, generación de informes, etc en Krptkn y la parte de extracción de metadatos en FITS.

3

Análisis de requisitos

Se dividen los requisitos del sistema en funcionales (funciones que debe cumplir) y no funcionales (características generales usadas para juzgar la solución).

Además se ha añadido una sección para comentar los no-requisitos. Estos son atributos en los que no se empleará tiempo de desarrollo por diversas razones (explicadas en esa sección).

3.1. Requisitos funcionales

Los siguientes son los requisitos funcionales de Krptkn:

- Crawling de páginas web.
- Extración de metadatos.
- Análisis de metadatos.
- Persistencia en base de datos.
- Generación de informes.

3.2. Requisitos no funcionales

El sistema a desarrollar deberá tener las siguientes características:

- Modular y extensible. Los distintos módulos por los que estará compuesto el sistema se podrán reemplazar sin cambios exteriores.
- Estabilidad. El sistema deberá poder correr durante largos periodos de tiempo, recuperarse de errores menores y reiniciarse en caso de que no se pueda recuperar.
- Escalable. En caso de ser necesario expandir la potencia de la solución, el escalado horizontal debe ser lo más simple posible.
- Facilidad de uso. Es importante que el sistema sea fácilmente manejado por una persona que no esté familiarizada con la tecnología.

3.3. No requisitos

Es tan importante conocer los requisitos del proyecto como conocer los no-requisitos. Por ello, se listan ahora atributos que serán evitados en el desarollo de la solución:

- Seguridad. El sistema no está diseñado para ser expuesto a internet y por tanto no se empleará tiempo en auditar su seguridad ni endurecerlo.
- Alto rendimiento. Aunque no se pretende que la solución sea lenta, cuando se trata con servidores externos (no preparados para aguantar mucho estrés) el cuello de botella no suelen ser los procesos internos.
-

4

Desarrollo de la solución

4.1. Introducción y arquitectura

La idea inicial de cómo debería funcionar el software no ha cambiado desde las primeras iteraciones, a continuación se muestra en forma de pseudocódigo el núcleo del sistema:

```
# Datos de salida
urls_no_visitadas = []
urls_visitadas = []
metadatos = []

# Datos de entrada
urls_no_visitadas.push(input_usuario())

# Funcion principal
for url in urls_no_visitadas:
    # Descargamos los datos de la URL
    archivo = descargar(url)

    # Si es HTML, extraemos URLs del archivo
    if is_html(archivo):
        nuevas_urls = extraer_urls(archivo)
        urls_no_visitadas.push(nuevas_urls)
    # Si no es HTML, extraemos metadatos
    else:
        nuevos_metadatos = extraer_metadatos(archivo)
        metadatos.push(nuevos_metadatos)
```

Aunque el núcleo del sistema es ese, necesitamos varios mecanismos para cumplir los distintos requisitos propuestos. También existen varias partes del núcleo que al ser escritos en Elixir se ven

transformados. Tanto ‘urls_no_visitadas‘ como ‘urls_visitadas‘ se convierten en dos tablas de ETS (Erlang Term Storage). Esto nos permite tener una base de datos en memoria similar a Redis con la ventaja de que no es necesario un servicio externo. Por otro lado, ‘metadatos‘ no se guarda en una tabla, entra directamente a la base de datos externa para su uso posterior. También es externo ‘extraer_metadatos‘ ya que como se verá en la sección de extracción de metadatos, se ha implementado una interfaz que llama a código nativo escrito en C.

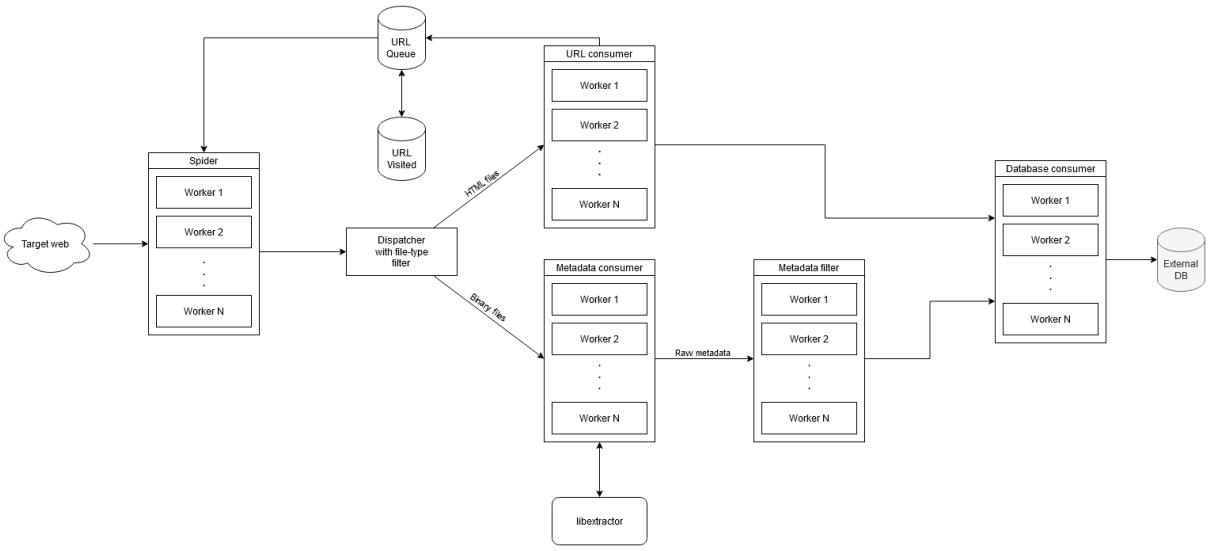

Figura 4.1: Vista de águila de la arquitectura de Krptkn

4.2. Creación de una araña

Como ya hemos visto, en el núcleo de la aplicación se encuentra la araña, que se encarga de explorar la página web en profundidad. El funcionamiento básico de la araña consiste en extraer una nueva URL de la tabla de URLs (URL Queue en la figura 4.1), hacer una petición GET al servidor y mandar el resultado al filtro que se ocupará de redirigir el contenido al módulo adecuado según sea HTML u otro tipo.

Existen dos casos en los que se pueden producir errores (siempre internos y controlados, no causan un cuelgue). En caso de que no exista ninguna URL en la tabla, se producirá un timeout y se reintentará hasta que alguien inserte una nueva URL a esta. También puede ocurrir que el servidor no responda a tiempo a la petición GET. Esto es ignorado y se trata de la misma forma que el problema anterior.

La librería que se usa para las peticiones HTTP es HTTPoison, inspirada por otra librería de Erlang llamada HTTPotion y usando como base Hackney. La decisión de usar HTTPoison se basa en que es fácil de usar, es madura y muy estable, usa SSL por defecto y no nos va a crear un cuello de botella.

Aunque algunos usuarios [1] [2] han notado problemas de escalado para aplicaciones de mucho tráfico en Hackney, he optado por ignorarlos ya que el cuello de botella de Krptkn es la parte de análisis de metadatos.

4.2.1. Parseo de HTML y limpieza de URLs

El módulo de parseo de HTML utiliza la librería Floki [3] para facilitar la búsqueda los distintos nodos del HTML. Floki ofrece tres parsers distintos:

- mochiweb_html: Escrito en Erlang, sin dependencias externas, lento.
- fast_html: Usa lexbor [4], escrito en C, muy rápido.
- html5ever: Usa html5ever [5], escrito en Rust, muy rápido también.

Se ha optado por el uso de fast_html por que ha sido el que menos fallos ha dado al probarlo con páginas web complejas. También es importante mencionar que la velocidad no era un problema en ningún caso, al usar muchos hilos paralelos, los tres cumplen de sobra el requisito de velocidad.

Una vez hemos parseado el HTML se buscan los nodos que contienen una propiedad "href.º "srcza que son los que más a menudo contienen URLs interesantes. Una vez obtenidos estos nodos, se pasa a la limpieza de las URLs.

La limpieza de URLs es heurística pero ha sido probada en un gran número de webs y no da errores. La heurística agrega o elimina barras oblicuas dependiendo de si el link lleva a un archivo o a otra página. Si el link encontrado no está dentro de los dominios que queremos visitar se filtra. También se ocupa de añadir el dominio base si este no se encontraba en la URL (/home"se convierte en "https://example.org/home"). Por último se filtran los links que usan otros protcolos como FTP, SSH, IPFS, etc.

Una vez se ha hecho esto, se eliminan los duplicados y el resto se insertan en la cola de URLs. La cola de URLs se ocupa de comprobar si ya hemos visitado el link que le hemos pasado.

4.3. Extracción de metadatos

[1] https://blog.appsignal.com/2020/07/28/the-state-of-elixir-http-clients.html
[2] http://big-elephants.com/2019-05/gun/
[3] https://github.com/philss/floki
[4] https://github.com/lexbor/lexbor
[5] https://github.com/servo/html5ever

4.3.1. Libextractor

De las distintas alternativas posibles para la extracción de metadatos se decidió usar libextractor por multiples razones:

- Lenguaje: Al ser una librería escrita en C, estamos seguros de que la máquina virtual de Erlang va a ser compatible. También es tranquilizador saber que si fuera necesario, puedo modificar el código fuente de forma relativamente fácil y rápida.
- Plugins: Libextractor está pensado para ser expandido con nuevos formatos. Cada formato que puede leer esta aislado en un plugin externo que se encarga de parsear el archivo y devolver los metadatos encontrados. Esto nos permite expandir las capacidades de Krptkn o quitar plugins que no sean necesarios o estén anticuados.
- Procesos aislados: Una característica importante del funcionamiento de libextractor es que las llamadas a los plugins externos son usando $fork$. Esto nos asegura que lo peor que puede ocurrir cuando intentamos extraer metadatos de un archivo corrupto o maligno es que el proceso del plugin muera y no nos devuelva metadatos.

A pesar de que estas características son muy importantes, libextractor también tiene multiples inconvenientes:

- Lentitud: A pesar de estar escrito en C, el cargar los plugins, crear un fork y copiar los datos del archivo a analizar puede ser demasiado lento en comparación con el spider de Krptkn. Debido a esto, la velocidad de exploración de Krptkn se ve limitada.
- Paralelismo: Una de las principales ventajas de el uso de Erlang/Elixir y OTP es su capacidad para paralelizar el código de forma muy simple. El estar usando una librería nativa que hace uso de forks y lee librerías compartidas (.so, .a, .o) hace imposible aprovechar el paralelismo que se usa en el resto de módulos de Krptkn..

Estos problemas se podrían eliminar a la vez que mantenemos las ventajas haciendo uso de una librería de extracción de metadatos escrita en Elixir. Esto nos permitiría hacer uso de tantos procesos paralelos como fuera necesario y acelerando mucho la ejecución. En las fases iniciales de desarrollo se intentó crear un módulo para leer los metadatos de archivos PNG y se concluyó que aunque la creación del parser había sido bastante rápida (entre dos y tres horas), desarrollar los módulos necesarios para igualar a libextractor (pdf, word, mp3, ogg, avi, deb, etc.) iba a ser imposible sin un equipo y más tiempo.

En conclusión, lo más adecuado ahora mismo es usar libextractor y si fuera posible, ir desarrollando parsers en Elixir para acelerar la fase de extracción. Ambos pueden ser usados a la vez gracias a la arquitectura de Krptkn por lo que el desarrollo de un parser para archivos que sabemos que son muy comunes (pdf o word) podría acelerar mucho el sistema sin emplear demasiado tiempo.

4.3.2. Filtros

Aunque en iteraciones iniciales del proyecto se había planteado la posibilidad de usar inteligencia artificial para la clasificación y filtrado de metadatos, se ha optado por una solución menos compleja: regex.

A través de unas reglas de regex muy simples, se extraen correos electrónicos y se comprueba

si el texto de los metadatos contiene o no palabras clave que son indicativas de que los metadatos contienen información sensible.

Los datos filtrados no son descartados. Los datos que pueden ser peligrosos se marcan como tal y se insertan en la base de datos para un posterior análisis estadístico y para la creación de los informes pertinentes.

4.3.3. Base de datos

La decisión principal con respecto a bases de datos que se toma en el desarrollo de este proyecto es sobre el uso de SQL o NoSQL. Debido a que los metadatos de diferentes formatos tienen estructuras muy distintas se suelen representar como pares de clave valor como $'Size' : '480x360'$. Esto encaja muy bien con bases de datos como MongoDB. A pesar de esto, se ha decidido usar PostgreSQL y usar en la columna que contiene los metadatos de cierto archivo el formato JSON. Podríamos haber usado una columna de texto pero el formato nativo JSON de PostgreSQL nos permite verificar que el formato es correcto. También tenemos la ventaja de poder cambiar de JSON a JSONb en cualquier momento. JSONb (json binario) nos daría una mayor velocidad de procesamiento a cambio de perder velocidad de inserción. Y por último, ya que estoy más familiarizado con SQL y el objetivo del proyecto no es aprender sobre NoSQL, usar PostgreSQL es la decisión más segura y que menos problemas va a dar.

4.4. Interfaz web

Una vez la funcionalidad básica del proyecto estaba terminada, se empezó a desarrollar una interfaz web para facilitar el uso por parte de usuarios sin experiencia. También ha resultado extremadamente útil para encontrar problemas en la aplicación.

La decisión de usar un framework web en lugar de una interfaz de terminal o una interfaz de usuario nativa se debe a que Krptkn está pensado para ser ejecutado en un servidor sin interfaz gráfica y para este tipo de despliegues lo más adecuado es una interfaz web a la que puedas acceder desde un ordenador en la misma red local. También se podría configurar para su uso a través de internet aunque el software no esta pensado para aceptar múltiples usuarios a la vez.

El framework web que utiliza Krptkn es Phoenix. Este es el estándar para aplicaciones web escritas en Elixir y es también con el que estoy más familiarizado. Aunque en un principio se consideró la idea de usar la variante LiveView de Phoenix que hace uso de websockets para actualizar la interfaz sin necesitar código Javascipt, finalmente se ha decidido usar Phoenix en conjunto con APIs JSON para actualizar los datos de la interfaz. Esto nos da más flexibilidad en el futuro para añadir nuevas funciones y aumenta mucho la velocidad de desarrollo de la parte web (en la cual no quería invertir demasiado

tiempo).

5

Resultados

6

Conclusiones y trabajo futuro

Debido a la enorme extensión que puede adquirir un sistema que realiza las tareas propuestas, es difícil decir que está terminado. A pesar de esto, se han cumplido tanto las tareas propuestas para un producto mínimo viable (spider, extracción de metadatos, creación de informes, etc.) como los extras (la interfaz web, el panel de control, el diseño, etc.). Por ello, los resultados han sido en general satisfactorios.

Aún siendo el resultado bueno, quedan múltiples áreas en las que se puede trabajar para tener un producto realmente profesional:

- Añadir un sistema automatizado de despliegue. Un pipeline en el que se pudieran integrar el testeo y despliegue a través de un repositorio Git sería enormemente útil para darle un uso profesional a la herramienta.
- Soporte para distintas bases de datos. Sería de especial utilidad para CI/CD el poder utilizar SQLite por ejemplo.
- Soporte para múltiples usuarios en la interfaz web. Gracias a Phoenix/Elixir, esto no debería ser muy costoso de desarrollar y tener varios usuarios en cada despliegue podría ser muy importante en ciertos casos de uso.
- Extracción de metadatos: el núcleo del sistema. Todavía se pueden aplicar múltiples mejoras (discutidas en profundidad en el capítulo sobre libextractor).

Bibliografía

Definiciones

opción de estilo Son los valores que modifican el funcionamiento del estilo. Se ponen entre corchetes y separadas por comas en el comando \documentclass y antes de el nombre del estilo que irá entre llaves.

Acrónimos

WYSIWYG What You See Is What You Get.

WYTIWYG What You Think Is What You Get.

Apéndices

A

WORD® vs. LATEX 2ε

A.1. Ventajas e inconvenientes de LATEX 2ε

El gusto por el LATEX depende de la forma de trabajar de cada uno. La principal virtud es la facilidad de formatear cualquier texto y la robustez. Incluir referencias a capítulos, secciones, figuras, tablas, etc. es inmediato. Las ecuaciones quedan estupendamente, la escritura se puede realizar modularizada y estructurada. Con estilos como el presentado se reduce considerablemente la utilización de paquetes complejos reduciendo su uso a comandos simples aunque liminatos

El principal inconveniente de LATEX 2ε radica en la necesidad de aprender un conjunto de comandos para generar los elementos que queremos. Cuando se está acostumbrado a un entorno "lo que veo es lo que obtengo" (WYSIWYG) es difícil cambiar la mentalidad a un entorno del tipo "lo que pienso es lo que obtengo" (WYTIWYG) como LATEX 2ε.

Por otro lado, en general será muy complicado cambiar el formato para desviarnos de la idea original de sus creadores del estilo. No es imposible, pero sí muy difícil. En muchos casos, comno en el tipo de documentos a los que está dirigido este estilo de LATEX 2ε es una ventaja y no un inconveniente en el caso de querer obtener una imagen corporativa en los documentos.

A.2. Ventajas e inconvenientes de Word®

La ventaja mayor del Word® es que permite configurar el formato muy fácilmente. Para las ecuaciones tradicionalmente ha proporcionado pésima presentación. Sin embargo, el software adicional Mathtype® solventa este problema, incluyendo una apariencia muy profesional y cuidada. Incluso permite utilizar un estilo similar al de LATEX 2ε. Además, aunque el Word® incluye sus propios atajos para escribir ecuaciones, Mathtype® admite también la escritura de ecuaciones utilizando los mismos comandos que LATEX 2ε.

Trabajar con títulos, referencias cruzadas e índices es un engorro, por no decir nada sobre la creación de una tabla de contenidos. Resulta muy frecuente que alguna referencia quede pérdida o huér-

fana y aparezca un mensaje en negrita indicando que no se encuentra. Algunos autores hacen todas estas referencias manualmente lo significa que cualquier cambio supone un arduo trabajo rehaciendo las referencias de todo el documento. Los estilos permiten trabajar bien definiendo la apariencia, pero también puede desembocar en un descontrolado incremento de los mismos. Además, es muy probable que Word® se quede colgado, sobre todo al trabajar con copiar y pegar de otros textos y cuando se utilizan ficheros de gran extensión, como es el caso de un libro.

A.3. ¿Cuál elijo?

En cualquier caso las tipografías, colores, distancias entre párrados, interlineados, encabezamientos y estructura del docuemnto debe coincidir con el aquí presentado. Dado que sólo se aporta el estilo de LaTeX 2ε se recomienda su uso aunque no es obligatorio y es decisión del autor elegir.

B

Instalación

B.1. Linux

Este paquete en Linux puede instalarse de tres formas diferentes. La primera de ella es a través de los sistemas de paquetes usados habitualmente en Linux, para ello basta con configurar el repositorio con los siguientes comandos: **sudo deb http://metis.ii.uam.es asignaturas main**. Una vez configurado es necesario validarlo obteniendo la clave pública del repositorio. Para ello basta con acceder a uno de los servidores del anillo de claves de ubuntu para obtenerla. La forma de hacerlo es ejecutando como superusuario el siguiente comando: **sudo apt-key adv --keyserver keyserver.ubuntu.com --recv C95B8FCEC5A57017**. Para finalizar basta con ejecutar: **sudo apt-get update && sudo apt-get install tfgtfmthesisuam**. Una vez realizados todos estos comandos el paquete estará instalado y funcional y con acceso a las actualizaciones que se realicen sobre el paquete. En este caso la documentación estará en el directorio /usr/share/doc/tfgtfmthesisuam.

La segunda forma es descargar el fichero tfgtfmthesisuam.deb de la página moodle correspondiente e instalarlo con el comando **sudo dpkg -i tfgtfmthesisuam.deb** estando en el mismo directorio en el que se ha descargado el fichero. Sin embargo esta opción no es tan sencilla dado que si no se tienen los paquetes de los que depende este paquete producirá un error hasta que estén todos instalados. Además, si se instala de esta forma no está disponible el acceso a actualizaciones de forma automática. En este caso la documentación estará en el directorio /usr/share/doc/tfgtfmthesisuam.

La tercera posibilidad consiste en descargar el fichero .tgz o el .zip del estilo y descomprimirlo en cualquier directorio. Si se ejecuta **sudo installLinux** el estilo será instalado en el sistema y estará disponible desde cualquier directorio y usuario y puede eliminarse del directorio donde ha sido descomprimido. Si no se tiene acceso como superusuario se puede no ejecutar el comando pero entonces es necesario que el estilo y el documento se encuentren en el mismo directorio y que además sea el mismo directorio desde el que se compila. Si se utiliza este método no se tiene acceso a las actualizaciones automáticas. En este caso la documentación estará en el directorio /usr/share/doc/tfgtfmthesisuam.

B.2. Windows™

Es necesario instalar MiKTeX completo e instalar manualmente algunos paquetes así como este estilo.

Las indicaciones que se presentan a continuación no han sido probadas y sólo son indicaciones que teóricamente deberían funcionar pero si no lo hacen, el creador de este estilo agradecería que se comunicase qué instrucción no funciona.

1.– Crear el directorio c:\localtexmf como administrador de Windows.

2.– Descomprimir el estilo zip en ese directorio.

3.– Activar dicho directorio como directorio de estilos de latex para ello es necesario utilizar una de las siguientes dos opciones:

3.1.– Usando el GUI de MiKTeX:

3.1.1.– En el menú Inicio ve a la entrada MiKTeX y abre la configuración "Configuración (Administrador)", por supuesto. Se abrirá la ventana "Opciones MiKTeX".

3.1.2.– Ve a la pestaña "Raíces". Haz clic en "Añadir" y elije c:\localtexmf.

3.1.3.– Ahora la parte más importante: ve a la pestaña "General" y haz clic en "Actualizar FNDB" (FNDB = File Name Data Base). En algunos casos, especialmente si hay nuevas fuentes instaladas, hay que pulsar también el botón "Actualizar Formatos".

3.2.– En la línea de comandos (siempre añadiendo –admin para actuar como administrador y opcionalmente –verbose):

3.2.1.– Ejecuta `initexmf –register-root=c:\localtexmf`

3.2.2.– Ejecuta `initexmf –update-fndb`

Dado que no dispongo de ningún ordenador en este sistema operativo este apartado se actualizará adecuadamente en el momento en el que algún estudiante me comunique cómo lo ha realizado o me solicite ayuda para instalarlo.

B.3. Mac OS X

En el caso de querer instalar el estilo en este sistema es necesario instalar MacTex. Para ello se puede ir a la página oficial de MacTex pinchando http://tug.org/mactex y seguir las instrucciones correspondientes con todas las actualizaciones necesarias. Dependiendo de la versión de MacTex este se instala en el directorio /usr/local/texlive/XXXX donde XXXX es el año de la versión de MacTex que se esté instalando y cuyo valor es necesario saber para instalar correctamente el estilo.

En Mac OS X es necesario descargar el fichero .tgz o el .zip del estilo y descomprimirlo en cualquier directorio. Si se ejecuta **sudo installMaC XXXX** el estilo será instalado en el sistema y estará disponible desde cualquier directorio y usuario y puede eliminarse del directorio donde ha sido descomprimido. Si

no se tiene acceso como superusuario se puede no ejecutar el comando pero entonces es necesario que el estilo y el documento se encuentren en el mismo directorio y que además sea el mismo directorio desde el que se compila. Cualquier actualización debe realizarse manualmente realizando el mismo procedimiento.

B.4. Overleaf o ShareLatex

Para utilizar este estilo en alguno de estas aplicaciones web es necesario bajarse el archivo .tgz o .zip, descomprimirlo y subir los ficheros tfgtfmthesisuam.cls, tfgtfmthesisuam.ist, y todas las imágenes. Pueden borrarse todos los logos e imágenes que no se correspondan con la escuela o facultad correspondiente. En general estos sistemas en su versión gratuita tienen limitado el número de archivos que se pueden subir por cada proyecto y por tanto es necesario no desperdiciar espacio con ficheros innecesarios, sobre todo si se va a estructurar mucho el documento o se utilizan muchas imágenes o fuentes de código. Si se dispone de una cuenta de pago en alguno de estos sistemas entonces hay muchas menos limitaciones y se pueden copiar todos los ficheros.

B.5. ¿Dónde está el manual?

Dónde se encuentre el manual depende mucho del sistema operativo y el tipo de instalación realizada por ello se recomienda que se busque el fichero tfgtfmthesis.pdf. En ese mismo directorio se encontrarán los fuentes del manual a los que se hace referencia a lo largo de todo este documento.

B.6. Corrección ortográfica y codificación de caracteres

La corrección ortográfica depende exclusivamente del editor que se esté utilizando y por tanto es necesario acudir a la documentación del editor que se esté utilizando para configurarla correctamente.

Por otro lado todo el estilo se ha creado utilizando la codificación UTF8 y por tanto la codificación de los fuentes debe estar también en UTF8. Debe seleccionarse dicha codificación en el editor que se esté utilizando.

B.7. ¿Qué editor utilizo?

En todos los sistemas operativos hay múltiples editores de LaTeX 2_ε e incluso algunos entornos de desarrollo integrados como eclipse o netbins así como editores como vi, emacas, sublime o atom tienen

plugins o packetes que pueden ser instalados para que reconozcan la sintaxis de LaTeX 2_ε y pueden compilar los documentos. La elección depende de cada uno y depende de los gustos, habilidades y conocimientos de cada uno. Lo recomendable es probar con varios hasta encontrar el adecuado.

C

Packetes incluidos

Este estilo utiliza múltiples paquetes que se indican a continuación con enlaces a sus manuales en la WEB. Estos paquetes se utilizan con ciertos parámetros que se indican a continuación de cada elemento del listado. Para acceder a la documentación pincha en documentación en cada paquete.

algorithm2e (Documentación).

alltt (Documentación).

amsmath (Documentación).

babel (Documentación).

calc (Documentación).

caption small,bf,margin={5em,5em}. (Documentación).

cite (Documentación).

cleveref (Documentación).

etex (Documentación).

etoolbox (Documentación).

eurosym (Documentación).

fancyhdr (Documentación).

fancybox (Documentación).

filecontents (Documentación).

float (Documentación).

gnuplottex shell,cleanup,subfolder. (Documentación).

glossaries acronyms,nonumberlist,shortcuts,toc. (Documentación).

graphicx (Documentación).

hyperref (Documentación).

ifpdf (Documentación).

inputenc utf8. (Documentación).

lipsum (Documentación).

listings (Documentación).

longtable (Documentación).

makeidx (Documentación).

morewrites (Documentación).

multicol (Documentación).

multirow (Documentación).

pgf (Documentación).

pgfgantt (Documentación).

pgfplots (Documentación).

subfigure hang,TABBOTCAP. (Documentación).

tocloft subfigure. (Documentación).

twoopt (Documentación).

verbatim (Documentación).

wrapfig (Documentación).

xcolor (Documentación).

D

Resumen de opciones del estilo

En este apéndice se presentan las múltiples opciones de estilo (clase) y sus funciones de forma resumida aunque todas ellas ya han sido presentadas donde corresponde es aconsejable disponer de un resumen de ellas. Estas opciones son las siguientes:

- Tamaño de la página

  normalbook Opción por defecto, no hace falta indicarla. Utiliza A4 como tamaño de página.

  smallbook Con esta opción se utiliza el tamaño B5 como tamaño de página y se reduce la tipografía de acuerdo con la reducción de tamaño.

  tinybook Con esta opción se utiliza el tamaño C5 como tamaño de página, siendo el tamaño más pequeño posible y se reduce la tipografía de acuerdo con la reducción de tamaño.

- Idioma

  spanish Opción por defecto, no hace falta indicarla. Usa el español como lenguaje base. El abstract y el resumen se ordenan de acuerdo con ello.

  english Usa el idioma inglés como lenguaje base. El abstract y el resumen se ordenan de acuerdo con ello.

- Tipo de documento

  tfg Opción por defecto, no hace falta indicarla. El documento será un trabajo fin de grado.

  tfm El documento será un trabajo fin de máster.

  thesis El documento será una tesis.

- Tipo de copyright

  copyright Opción por defecto, no hace falta indicarla. Muestra el copyright normal en el reverso de la portada.

  copyleft Muestra el copyleft en el reverso de la portada.

  nocopyright No muestra ni copyright ni copyleft en el reverso de la portada.

- Tipo de índice.

  normalindex Opción por defecto, no hace falta indicarla. Muestra partes, capítulos y apartados.

  extendedindex Muestra también los subapartados. Es aconsejable pesnsar si en el documento creado tiene sentido o no mostrar ese nivel.

  fullindex Muestra hasta el nivel de subsubapartados. Su uso no se aconseja pero está disponible por si en algún caso se considera necesario dada la extensión del documento y la necesidad de mostrar en el índice la organización del documento hasta ese nivel.

- Listados de figuras, ecuaciones, algoritmos ...

  loall Muestra todos los listados.

  nonelo No muestra ningún listado.

  loa Muestra el listado de algoritmos y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de algoritmos en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

  loc Muestra el listado de códigos y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de códigos en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

  loe Muestra el listado de ecuaciones y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de ecuaciones en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

  lof Muestra el listado de figuras y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de figuras en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

  lot Muestra el listado de tablas y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de tablas en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

  lotb Muestra el listado de cuadros de texto y es compatible con cualquier otra opción menos con loall y nonelo. Se recomienda que no se utilice si el número de cuadros de texto en todo el documento es menor de tres salvo que se considere adecuado hacerlo.

- Aspecto global

  covers Presenta las cubiertas institucionales.

  final Se prepara para la versión final eliminando la marca de agua.

  printable Por defecto se compila el documento con márgenes simétrico y esta opción lo que hace es añadir margen en la parte derecha de las páginas pares y en la izquierda de las páginas impares para dejar espacio para la encuadernación. Se recomienda una compilación con esta opción para la entrega electrónica y otra con ella para la entrega en papel.

  firstnumbered Por defecto las primeras páginas de cada capítulo no se numeran. Si se desea que sean numeradas debe ponerse esta opción.

- Imagen institucional

  epsbased Dado que este estilo ha sido diseñado inicialmente es la opción por defecto y no hace falta indicarla. Utiliza los colores institucionales de la Escuela Politécnica Superior y los logos y textos adecuados.

  uambased Es una opción genérica que utiliza los colores institucionales de la Universidad Autónoma de Madrid. Es necesario indicar el logo, su ancho, la facultad y cuantas variables sean necesarias para el correcto formato del documento.

  cienciasbased Utiliza los colores institucionales de la Facultad de Ciencias y los logos y textos adecuados.

  economicasbased Utiliza los colores institucionales de la Facultad de Ciencias Económicas y Empresariales y los logos y textos adecuados.

  derechobased Utiliza los colores institucionales de la Facultad de Derecho y los logos y textos adecuados.

filosofiabased Utiliza los colores institucionales de la Facultad de Filosofía y Letras y los logos y textos adecuados.

medicinabased Utiliza los colores institucionales de la Facultad de Medicina y los logos y textos adecuados.

enfermeriabased Utiliza los colores institucionales de las Escuelas de Enfermería, sin embargo al haber varias y los logos y textos deberán ser modificados utilizando las funciones adecuadas.

fisioterapiabased Utiliza los colores institucionales de las Escuelas de Fisioterapia, sin embargo al haber varias y los logos y textos deberán ser modificados utilizando las funciones adecuadas.

- Colores globales. Dado que sólo controlan los colores del documento es necesario utilizar todos los comandos necesarios para indicar los textos y logos del documento.

  blackbased Todas las decoraciones estarán en negro.

  graybased Todas las decoraciones estarán en tonos de gris.

  redbased Todas las decoraciones estarán en rojo.

  greenbased Todas las decoraciones estarán en verde.

  bluebased Todas las decoraciones estarán en azul.

  yellowbased Todas las decoraciones estarán en amarillo.

  magentabased Todas las decoraciones estarán en magenta.

  cyanbased Todas las decoraciones estarán en cian.

  orangebased Todas las decoraciones estarán en naranja.

- Control de compilación

  overleaf Permite compilar en overleaf pero no se dispone de xindy que es quien permite una ordenación de los acrónimos y las definiciones teniendo en cuenta que las letras con acento se ordenan con las letras sin acento. No se podrá utilizar tampoco la opción gnuplot.

  gnuplot Permite compilar código gnuplot.

E

Funciones y entornos

En este apéndice se presenta un listado de funciones y otro de los entornos. Se presentan en formato de tabla a modo de resumen y para que puedan ser impresas a modo de referencia. En cada función o entorno se indica si debe usarse en el cuerpo del texto o en el preámbulo, los parámetros opcionales si tiene y los obligatorios si tiene. También se presentan aquellos comandos de LaTeX 2$_\varepsilon$ que no sólo ha sido modificado su comportamiento sino que también han cambiado los parámetros.

E.1. Comandos en el preámbulo

Autoría		
\advisor		{tutor}
\author		{titulo}
\coadvisor		{cotutor}
\copyrightdate		{fecha}
\faculty		{facultad/escuela}
\levelin		{titulacion}
\title	[título corto]	{titulo}
\speaker		{ponente}

Decoraciones	
\coverdata	{texto}
\facultylogo	{fichero}
\facultylogowide	{dimension}

Directorios	
\codesdir	{directorio}
\datadir	{directorio}
\graphicsdir	{directorio}
\logosdir	{directorio}

Prefacio	
\abstractfile	{fichero}
\ackfile	{fichero}
\dedication	{dedicatoria}
\famouscite	{cita}
\keywords	{palabras}
\palabrasclave	{palabras}
\prefacefile	{fichero}
\privateaddress	{direccion}
\resumenfile	{fichero}

E.2. Commandos en el cuerpo del texto

Código		
\Code	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}{lenguaje}
\AdaCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\ASMCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\ASMMotorolaCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\CCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\CPPCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\CSharpCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\GnuplotCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\HaskellCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\HTMLCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\JavaCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\LaTeXCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\LispCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\MakeCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\MathematicaCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\MatlabCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\OctaveCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\PascalCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\PerlCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\PHPCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\PythonCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\RCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\RubyCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\ScilabCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\SQLCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\VHDLCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}
\XMLCode	[etiqueta]	{pie corto}{pie largo}{fichero}{línea inicial}{línea final}{numeración inicial}

Ecuaciones	
\boxed	{ecuación}

Estructura		
\cleardoublepage		
\part	[título corto]	{titulo}{fichero}
\part	[título corto]	{titulo}
\chapter	[título corto]	{titulo}{fichero}
\chapter	[título corto]	{titulo}
\section	[título corto]	{titulo}{fichero}
\section	[título corto]	{titulo}
\subsection	[título corto]	{titulo}{fichero}
\subsection	[título corto]	{titulo}
\subsubsection	[título corto]	{titulo}{fichero}
\subsubsection	[título corto]	{titulo}
\paragraph	[título corto]	{titulo}{fichero}
\paragraph	[título corto]	{titulo}
\subparagraph	[título corto]	{titulo}{fichero}
\subparagraph	[título corto]	{titulo}

Figuras y tablas		
\subfigure	[etiqueta]	{pie}{contenido}
\subtable	[etiqueta]	{pie}{contenido}

Gantt		
\milestone	[etiqueta enlace]	{etiqueta gantt}{tiempo}
\taskbar	[etiqueta enlace][porcentaje fin]	{etiqueta gantt}{tiempo inicio}{tiempo fin}
\taskgroup	[etiqueta enlace][porcentaje fin]	{etiqueta gantt}{tiempo inicio}{tiempo fin}
\FtoFlink		{eetiqueta inicio}{eetiqueta fin}
\FtoSlink		{eetiqueta inicio}{eetiqueta fin}
\StoSlink		{eetiqueta inicio}{eetiqueta fin}

Gráficas		
\plotlined		{fichero datos}{titulo datos}
\plotdatalined		{fichero datos}{titulo datos}
\plotdata		{fichero datos}{titulo datos}
\plotfunction		{fichero datos}{titulo datos}

Imágenes		
\image		{ancho}{alto}{fichero}
\imageIL	ancho	fichero

Presupuestos	
\budgettitle	{titulo presupuesto}
\concept	{título}{precio unitario}{cantidad}{coste total}
\separator	
\subconcept	{título}{precio unitario}{cantidad}{coste total}
\subtotal	{subtotal}
\total	{total presupuesto}

Texto citado	
\onlinecitation	{autor}{texto}

E.3. Entornos

De igual forma en la siguiente tabla se presentan los distintos entornos creados o modificados para este estilo. Todos los entornos deben usarse dentro del cuerpo del documento.

General		
algorithm	[pie corto]	{etiqueta}{pie completo}
algorithmN	[pie corto]	{etiqueta}{pie completo}
budget		
equation	[etiqueta]	{título}
figure	[pie corto]	{etiqueta}{pie completo}
gantt		{tiempo inicio}{tiempo fin}
largecitation		{autor}
multiequation		
table	[pie corto]	{etiqueta}{pie completo}
textbox	[pie corto]	{etiqueta}{pie completo}

Listados especiales
functionality
objetive
functional
nonfunctional
simplelist

Gráficas		
gnuplot	[opciones]	
loglogplot	[posicion leyenda]	{título}{título eje x}{título eje y}{ancho}{alto}
semilogxplot	[posicion leyenda]	{título}{título eje x}{título eje y}{ancho}{alto}
semilogyplot	[posicion leyenda]	{título}{título eje x}{título eje y}{ancho}{alto}
xyplot	[posicion leyenda]	{título}{título eje x}{título eje y}{ancho}{alto}

Índice terminológico


UAM
UNIVERSIDAD AUTONOMA
DE MADRID