



GRADO EN INGENIERÍA
MULTIMEDIA



VNIVERSITAT
DE VALÈNCIA

TRABAJO FIN DE GRADO

PLATAFORMA WEB PARA EL EMPAREJAMIENTO
DE PROFESIONALES Y PROYECTOS BASADA EN
COMPETENCIAS

AUTOR: JAVIER LEÓN SOLER

TUTOR: RAÚL PEÑA ORTIZ

JULIO 2025



VNIVERSITAT
DE VALÈNCIA



Escola Tècnica Superior
d'Enginyeria **ETSE-UV**

TRABAJO FIN DE GRADO

PLATAFORMA WEB PARA EL EMPAREJAMIENTO DE PROFESIONALES Y PROYECTOS BASADA EN COMPETENCIAS

AUTOR: JAVIER LEÓN SOLER

TUTOR: RAÚL PEÑA ORTIZ

Declaración de autoría:

Yo, Javier León Soler, declaro la autoría del [Trabajo Fin de Grado \(TFG\)](#) titulado “Plataforma web para el emparejamiento de profesionales y proyectos basada en competencias” y que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual. El material no original que figura en este trabajo ha sido atribuido a sus legítimos autores.

Valencia, 17 de abril de 2025

Fdo: Javier León Soler

Resumen:

Este es el resumen de [TFG](#). Debe ser corto (máximo media página) y cubrir los aspectos principales del [TFG](#).

Abstract:

This is the abstract of the TFG. It must be short and cover the main aspects of the TFG.

Resum:

Aquest és el resum del [TFG](#). Ha de ser curt (màxim mitja pàgina) i cobrir els aspectes principals del [TFG](#).

Agradecimientos:

En primer lugar quiero agradecer a todos aquellos que me han apoyado durante todos estos años.

En segundo lugar...

Índice general

1. Introducción	23
1.1. Introducción	23
1.2. Motivación	23
1.3. Objetivos	23
1.4. Organización de la memoria	23
2. Estado del arte	25
2.1. Análisis de aplicaciones similares	25
2.2. Evaluación de tecnologías	25
3. Requisitos, especificaciones, planificación, coste, riesgos y viabilidad	27
3.1. Requisitos	27
3.1.1. Requisitos funcionales	27
3.1.2. Requisitos no funcionales	27
3.2. Especificaciones	27
3.3. Planificación y estimación de costes	27
3.4. Riesgos	28
3.5. Viabilidad	28
4. Análisis	29
4.1. Diagrama de casos de uso	29
4.2. Diagrama de secuencia	29
4.3. Diagrama de clases de primer nivel	29
5. Diseño	31
5.1. Arquitectura de componentes	31
5.2. Despliegue del sistema	31
5.3. Modelo de datos	31
6. Implementación y pruebas	33
6.1. Implementación	33

6.2. Pruebas unitarias	33
6.3. Pruebas funcionales	33
6.4. Pruebas de rendimiento	33
6.5. Pruebas de usabilidad	33
6.6. Pruebas de seguridad	34
7. Conclusiones	35
7.1. Revisión de costes	35
7.2. Conclusiones	35
7.3. Trabajo futuro	35
8. Bibliografía	37
A. Glosarios	39
A.1. Acrónimos y abreviaciones	39
A.2. Términos	39
B. Ejemplos del lenguaje de marcado L^AT_EX	41
B.1. Algunos ejemplos básicos de uso de L ^A T _E X	41
B.2. Figuras y subfiguras	41
B.3. Tablas	42
B.4. Listados	45
B.5. Acrónimos y términos	46
C. Ejemplos de uso de B^IB_TE_X como gestor de bibliografía	49
C.1. Ejemplos de referencias formales bien valoradas	49
C.2. Ejemplos de referencias con valoración intermedia	49
C.3. Ejemplos de referencias con valoración baja a evitar en la medida de lo posible	49



Índice de figuras

B.1. Esta es una figura que latex decide donde colocar (floating) en el documento.	42
B.2. Los cuatro diagramas en mosaico	42

Índice de tablas

B.1. Ejemplo de tabla 1	42
B.2. Web workload generators and grade in which main features are fulfilled . .	43
B.3. Ejemplo de tabla muy larga para una página	44

Índice de listados de código

B.1. Comandos bash para generar el fichero PDF	45
B.2. Comandos bash para generar el fichero PDF con arara	46
B.3. Ejemplo de listado en Python	46
B.4. Ejemplo de listado en Java	46
B.5. Ejemplo de listado en HTML 	46
B.6. Ejemplo de listado en CSS 	46

Capítulo 1

Introducción

1.1. Introducción

1.2. Motivación

1.3. Objetivos

1.4. Organización de la memoria

Capítulo 2

Estado del arte

En el presente capítulo se lleva a cabo un análisis del estado del arte con el fin de situar el proyecto en el contexto de soluciones existentes y tecnologías actuales. La plataforma web desarrollada en este trabajo tiene como objetivo facilitar el emparejamiento entre profesionales y proyectos mediante un sistema basado en competencias jerarquizadas. Por ello, resulta esencial estudiar qué herramientas similares existen ya en el mercado y cómo se enfrentan a problemas de emparejamiento, recomendación o gestión de talento y proyectos.

Primero se presentarán aplicaciones con funcionalidades relacionadas, analizando sus características principales, puntos en común con esta propuesta y diferencias clave. Este estudio permitirá identificar tanto aspectos que se consideran imprescindibles en una plataforma de este tipo, como oportunidades de mejora o innovación.

Posteriormente se realizará un análisis crítico de las tecnologías más relevantes para el desarrollo del sistema, justificando la elección final en cada caso a partir de criterios como la escalabilidad, facilidad de desarrollo, mantenimiento o compatibilidad entre componentes.

Por último, se listarán aquellas herramientas de soporte utilizadas durante el desarrollo del trabajo, como entornos de desarrollo, sistemas de control de versiones o editores de diagramas, las cuales han sido fundamentales para llevar a cabo el proyecto.

2.1. Análisis de aplicaciones similares

2.2. Evaluación de tecnologías

Capítulo 3

Requisitos, especificaciones, planificación, coste, riesgos y viabilidad

3.1. Requisitos

3.1.1. Requisitos funcionales

3.1.2. Requisitos no funcionales

Se debe optar por formular los requisitos de forma que se pueda conocer si se han alcanzado o no a la finalización del proyecto. Por ejemplo, es difícil valorar si el siguiente requisito funcional se alcanza o no: *El sistema debe retornar una respuesta en un tiempo razonable cuando tenga muchos usuarios concurrentes*. ¿Cuánto es un tiempo razonable?, ¿cuántos son muchos usuarios?. Sin embargo, si se formula de este otro modo: *El sistema debe retornar una respuesta en menos de un segundo cuando tenga 200 usuarios concurrentes*, es fácil comprobar si se ha alcanzado ejecutando un plan de pruebas, por ejemplo con JMeter.

3.2. Especificaciones

Especificación del proyecto a partir de los requisitos. Una vez se han definido los requisitos funcionales y no funcionales del sistema, se procede a especificar el sistema a partir de ellos. En este apartado se muestran características del sistema, sin entrar en detalles de diseño e implementación, que permiten entender el problema al que nos enfrentamos pero no como lo vamos a afrontar (etapa de diseño). Se recomienda confeccionar una arquitectura de referencia, o imagen no técnica (más bien comercial) que ilustre los actores y aplicaciones principales que conforman el sistema.

3.3. Planificación y estimación de costes

Describir el tipo de metodología de desarrollo que se va a utilizar (cascada, ágil, etc). Tareas a realizar, estimación de la duración de las tareas, y distribución temporal (por ejemplo con un diagrama de Gantt).

Enumerar las tareas a realizar, estimación de la duración de las tareas, y distribución temporal (por ejemplo con un diagrama de Gantt).

Costes de personal (teniendo en cuenta los costes de seguridad social), de hardware (imputando solo la duración del proyecto y teniendo en cuenta que los equipos se amortizan en 3 o 4 años) y/o de software. Además, hay que añadir costes indirectos.

3.4. Riesgos

Identificación de los riesgos que pueden aparecer durante el desarrollo del proyecto, su probabilidad de ocurrencia, su impacto en el proyecto y las medidas que se podrían adoptar para mitigarlos.

3.5. Viabilidad

En este apartado, dependiendo de la naturaleza del proyecto, se debería analizar la viabilidad técnica y la viabilidad económica. Para la viabilidad técnica hay que analizar si los recursos necesarios (herramientas, conocimientos, experiencia, etc) para llevar a cabo el proyecto permiten realizarlo en el tiempo previsto. En cuanto a la viabilidad económica hay que evaluar si el proyecto será rentable cuando esté operativo.

Capítulo 4

Análisis

4.1. Diagrama de casos de uso

Identificación de actores y diagramas de casos de uso asociados a los requisitos funcionales.

4.2. Diagrama de secuencia

Para los casos de uso más importantes y complejos se deben especificar los diagramas de secuencia que describan el comportamiento de los objetos que intervienen en el caso de uso.

4.3. Diagrama de clases de primer nivel

Al final del análisis se debe presentar un diagrama de clases de primer nivel que muestre las clases principales del sistema y sus relaciones. Para cada una de las clases que tengan un estado complejo o relevante, se debe especificar el diagrama de estados asociado o el diagrama de actividades en el que se involucra la clase.

Capítulo 5

Diseño

5.1. Arquitectura de componentes

A partir de la especificación realizada en la sección 3.2 del Capítulo 3 y la tecnología seleccionada en la sección 2.2 del Capítulo 2, se debe construir el diagrama de componentes UML que represente la arquitectura de referencia del sistema. En este diagrama se deben identificar los componentes principales del sistema y las relaciones entre ellos, pasando a detallar los subcomponentes de cada uno de ellos en secciones posteriores.

5.2. Despliegue del sistema

En esta sección se debe presentar el diagrama de despliegue del sistema, que muestre la arquitectura física del sistema y cómo se distribuyen los componentes en los nodos físicos o virtuales que lo componen. En otras palabras, para cada componente principal del sistema se identifica en que servidor físico o virtual se va a ejecutar.

Se puede definir un despliegue de desarrollo (el despliegue que se tiene en el portátil mientras se trabaja), uno de producción (despliegue final o real) y uno de pruebas (lo más parecido posible al real).

5.3. Modelo de datos

En este caso se puede especificar el diagrama de clases final del diseño atendiendo al lenguaje de programación y a la tecnología seleccionada. Además será necesario, como mínimo, especificar el modelo físico de datos, en el lenguaje SQL propio de la base de datos relacional seleccionada, o el modelo de documentos, si se utiliza una base de datos NoSQL.

Capítulo 6

Implementación y pruebas

Las secciones presentadas son orientativas y no representan necesariamente la organización que debe tener este capítulo.

6.1. Implementación

Presentar cómo se ha organizado el desarrollo de los proyectos (capturas del IDE), trozos de código relevantes, cómo han quedado implementadas las interfaces gráficas de usuario, etc.

6.2. Pruebas unitarias

Descripción de las pruebas que se han llevado a cabo para comprobar que el código desarrollado es correcto (JUnit, etc).

6.3. Pruebas funcionales

Descripción de las pruebas que se han llevado a cabo para comprobar que los casos de uso identificados funcionan correctamente.

6.4. Pruebas de rendimiento

Descripción de las pruebas de estrés realizadas para comprobar los tiempos de respuesta de la aplicación (según figuren en los requisitos).

6.5. Pruebas de usabilidad

Descripción de las pruebas que se han llevado a cabo con usuarios para determinar el nivel de usabilidad de la aplicación (que se hayan recogido en los requisitos).

6.6. Pruebas de seguridad

Descripción de las pruebas realizadas para comprobar que se cumplen las restricciones de autenticación y de autorización que se han descrito en los requisitos.

Capítulo 7

Conclusiones

7.1. Revisión de costes

Al finalizar el proyecto hay que ver en qué fases de la ejecución nos hemos desviado, explicar los motivos y calcular el coste real con estos ajustes.

7.2. Conclusiones

7.3. Trabajo futuro

Capítulo 8

Bibliografía

- [1] Raúl Peña Ortiz, “Accurate workload design for web performance evaluation,” Tesis doctoral, Universitat Politècnica de València, ene. de 2013, ISBN: 9788490480250. DOI: [10.4995/thesis/10251/21054](https://doi.org/10.4995/thesis/10251/21054) (citado en página 42).
- [2] Frank Mittelbach y Ulrike Fischer, *The L^AT_EX Companion*, 3.^a edición. Addison-Wesley, 2023, ISBN: 978-0138166489 (citado en página 49).
- [3] H. Frank Cervone, “Getting Started with Cloud Computing: A LITA Guide,” en Edward M. Corrado y Heather Lea Moulaison, edición. Neal-Schuman Publishers, Inc, 2011, capítulo Cloud Computing: Pros and Cons, páginas 30-31, ISBN: 978-1-55570-749-1 (citado en página 49).
- [4] Roy Thomas Fielding y Richard N. Taylor, “Principled Design of the Modern Web Architecture,” *ACM Transactions on Internet Technology*, volumen 2, número 2, páginas 115-150, mayo de 2002, ISSN: 1533-5399. DOI: [10.1145/514183.514185](https://doi.org/10.1145/514183.514185) (citado en página 49).
- [5] Roy Thomas Fielding y Richard N. Taylor, “Principled Design of the Modern Web Architecture,” en *Proceedings of the 22nd International Conference on Software Engineering*, época ICSE ’00, Limerick, Ireland: Association for Computing Machinery, jun. de 2000, páginas 407-416, ISBN: 1581132069. DOI: [10.1145/337180.337228](https://doi.org/10.1145/337180.337228) (citado en página 49).
- [6] Andoni Salcedo-Navarro, Raúl Peña-Ortiz, José M. Claver, Miguel Garcia-Pineda y Juan Gutiérrez-Aguado, “Cloud-Native GPU-Enabled Architecture for Parallel Video Encoding,” en *Euro-Par 2024: Parallel Processing*. Springer Nature Switzerland, ago. de 2024, páginas 327-341, ISBN: 9783031695834. DOI: [10.1007/978-3-031-69583-4_23](https://doi.org/10.1007/978-3-031-69583-4_23) (citado en página 49).
- [7] Roy Thomas Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” Tesis doctoral, University of California, Irvine, 2000. dirección: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf (citado en página 49).
- [8] Peter Mell y Tim Grance, “The NIST Definition of Cloud Computing,” NIST Special Publication 800-145, 2011. DOI: [10.6028/nist.sp.800-145](https://doi.org/10.6028/nist.sp.800-145) (citado en página 49).
- [9] Alexey Melnikov, “The WebSocket Protocol,” Internet Engineering Task Force (IETF), Standard, dic. de 2011. dirección: <https://tools.ietf.org/html/rfc6455> (citado en página 49).
- [10] Luis Thuillier Larena, “Sistema de control de aforos dinámico mediante dispositivos IoT,” Trabajo Fin de Grado, Escola Tècnica Superior d’Enginyeria - Universitat de València, 2021 (citado en página 49).

-
- [11] MA Awad, “A comparison between agile and traditional software development methodologies,” Trabajo Fin de Máster, University of Western Australia, 2005. dirección: <https://www.academia.edu/download/58993716/10.1.1.464.609020190422-13963-j0ju8a.pdf> (citado en página 49).
 - [12] Django Software Foundation, “Django Documentation,” Manual Release 4.1.4, jul. de 2023. dirección: <https://buildmedia.readthedocs.org/media/pdf/django/4.1.x/django.pdf> (citado en página 49).
 - [13] Microsoft, *Visual Studio Code*, 2023. visitado nov. de 2023. dirección: <https://code.visualstudio.com> (citado en página 49).

Apéndice A

Glosarios

A.1. Acrónimos y abreviaciones

SGBD

Sistema de Gestión de Base de Datos. [46](#)

TFG

Trabajo Fin de Grado. [5](#), [7](#), [9](#), [11](#), [49](#)

TFM

Trabajo Fin de Máster. [46](#), [49](#)

A.2. Términos

Middleware

Software que se encuentra entre el sistema operativo y las aplicaciones que funcionan sobre él. [46](#)

Apéndice B

Ejemplos del lenguaje de marcado L^AT_EX

B.1. Algunos ejemplos básicos de uso de L^AT_EX

Texto en el párrafo 1.

Texto en el párrafo 2.

Texto en el párrafo 3.

- Consideración 1
- Consideración 2

1. Punto 1

2. Punto 2

A continuación se muestra una ecuación:

$$\int_0^1 \frac{1}{x^2 + 1} dx$$

B.2. Figuras y subfiguras

Podemos incluir imágenes en formato: png, pdf o jpg.

En la Figura B.1 se muestra un diagrama¹, mientras que la Figura B.2 muestra varios diagramas en mosaico, de manera que puedo referirme al segundo diagrama como Figura B.2b.

¹Realizado con <https://www.yworks.com/products/yed>

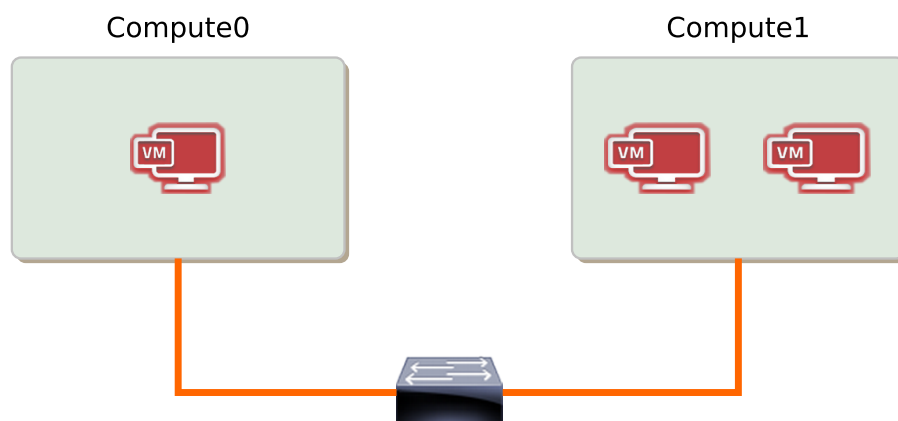


Figura B.1: Esta es una figura que latex decide donde colocar (floating) en el documento.

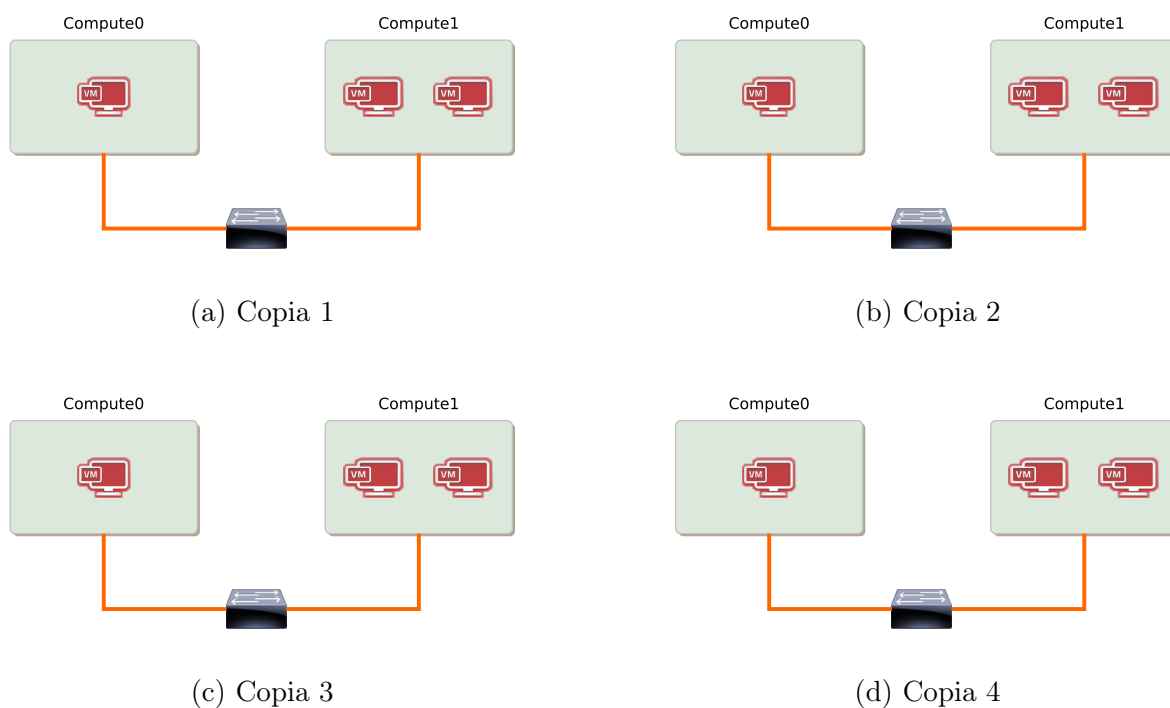


Figura B.2: Los cuatro diagramas en mosaico

B.3. Tablas

La Tabla B.1 es un ejemplo de una tabla.

Columna 1	Columna 2
1	2

Tabla B.1: Ejemplo de tabla 1

Existen también tablas más complejas que requieren un mayor control de la posición de los elementos, como la Tabla B.2 que ha sido extraída de [1], o de varias páginas, como la Tabla B.3.

GENERATOR FEATU./CAPAB.	GROUP I					GROUP II			GROUP III								
	WebStone	SPECweb	SURGE	Web Polyrgraph	TPC-W	LoadRunner	WebLOAD	JMeter	S-Clients	WebJamma	Deluge	HAMMERHEAD 2	PTester	Siege	HTTPERF	Autobench	
	★	★	★	★	★	★	★	★									
	★	★	★	★		★	★										
	★	☆		☆	★	★	★	★		★		☆			☆		
	★	★			★	★	★	★									
		★			★	★	★	★									
						★	★	★									
						★	★	★	★								
									★								★
		★		★	★		★	★		★	★	★	★	★	★	★	★
							★	★	★								
		★	★		★		★	★	★								
							★	★									
	User's Dynamism				★	★	★	★	★	★	★	★	★	★	★	★	★

★ Full support ☆ Partial support

Tabla B.2: Web workload generators and grade in which main features are fulfilled

Tabla B.3 – continued from previous page

[illegible]

B.4. Listados

Para generar el fichero PDF, podemos usar los comandos del Listado B.1.



```
1 pdflatex main.tex
2 bibtex main
3 pdflatex main.tex
```

Listado B.1: Comandos bash para generar el fichero PDF

También se puede usar `arara` que automáticamente regenera la bibliografía, ver los comandos del Listado B.2.

```
1 | arara main.tex
```

Listado B.2: Comandos bash para generar el fichero PDF con arara

También se pueden incluir listados de otros lenguajes como Python, ver Listado B.3, Java, ver Listado B.4, HTML , ver Listado B.5 o CSS , ver Listado B.6.

```
1 | def hello():
2 |     print("Hello World!")
```

Listado B.3: Ejemplo de listado en Python

```
1 | public class HelloWorld {
2 |     public static void main(String[] args) {
3 |         System.out.println("Hello, World");
4 |     }
5 | }
```

Listado B.4: Ejemplo de listado en Java

```
1 | <!DOCTYPE html>
2 | <html>
3 | <head>
4 | <title>Page Title</title>
5 | </head>
6 | <body>
7 |
8 | <h1>This is a Heading</h1>
9 | <p>This is a paragraph.</p>
10 |
11 | </body>
12 | </html>
```

Listado B.5: Ejemplo de listado en HTML 

```
1 | body {
2 |     background-color: lightblue;
3 | }
4 |
5 | h1 {
6 |     color: white;
7 |     text-align: center;
8 | }
9 |
10 | p {
11 |     font-family: verdana;
12 |     font-size: 20px;
13 | }
```

Listado B.6: Ejemplo de listado en CSS 

B.5. Acrónimos y términos

L^AT_EX tiene un paquete de glosario de términos y acrónimos que permite definir términos y acrónimos y usarlos en el texto. Lo que al principio puede parecer engorroso, una vez se usa, es muy útil.

Usamos por primera vez el acrónimo Trabajo Fin de Máster (TFM) y luego volvemos a usar el acrónimo TFM, que ahora aparece abreviado, y por primera vez el término *Middleware*.

Tenemos una opción especial para la forma plural de los acrónimos, como Sistema de Gestión de Base de Datos (SSGGBDD), aunque luego se usen en singular, SGBD.

Podemos forzar que aparezca el término completo, Trabajo Fin de Grado, abreviado, TFG, o ambos, Trabajo Fin de Grado (TFG). Aunque en estos casos no nos lleva a la tabla de acrónimos y términos.

El fichero `acronimos-terminos.tex` contiene ejemplos de definición de acrónimos y términos.

Apéndice C

Ejemplos de uso de BIBTEX como gestor de bibliografía

Este apéndice muestra ejemplos de uso de BIBTEX como gestor de bibliografía.

C.1. Ejemplos de referencias formales bien valoradas

- Un libro [2] o un capítulo de libro [3].
- Un artículo de revista [4].
- Una publicación en una conferencia [5] o una publicación de conferencia que forma parte de una colección [6].
- Una tesis doctoral [7].
- Un informe técnico específico del NIST [8].

C.2. Ejemplos de referencias con valoración intermedia

- Un informe técnico genérico [9].
- Un TFG [10] o un TFM [11].

C.3. Ejemplos de referencias con valoración baja a evitar en la medida de lo posible

- Un manual online [12].
- Una página web [13].