Un dibujo de un perro

Descripción generada automáticamente con confianza media

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

GRADO EN INGENIERÍA EN INFORMÁTICA

(Título)

(Nombre del Autor)

Un dibujo de un perro

Descripción generada automáticamente con confianza media

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

BACHELOR IN COMPUTING ENGINEERING

**(**Title**)**

(Author’s Name)

Month, Year

Un dibujo de un perro

Descripción generada automáticamente con confianza media

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

Departamento de (departamento del tutor académico)1

[Tecnología específica]2

(Título)

Autor: (Nombre del Autor)

Tutor académico: (Nombre del Tutor Académico) Cotutor académico: (Nombre del Cotutor Académico)

Mes, Año

1 Tecnologías y Sistemas De Información // Matemáticas // Ingeniería Eléctrica, Electrónica, Automática y Comunicaciones // o cualquier otro departamento de la UCLM al que pertenezca el tutor académico.

2 Computación, Ingeniería de Computadores, Ingeniería del Software o Tecnologías de la Información

Un dibujo de un perro

Descripción generada automáticamente con confianza media

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

Department of (departamento del tutor académico)3

[Specialization]4

(Title)

Author: (Nombre del Autor)

Supervisor: (Nombre del Tutor Académico)

Co-supervisor: (Nombre del Cotutor Académico)

Month, Year

3 Technologies and Information Systems // Mathematics // Electrical and Electronic Engeneering, Automatic Control and Communications // o cualquier otro departamento de la UCLM al que pertenezca el tutor académico.

4 Computing, Computer Engineering, Software Engineering or Information Technologies

**TRIBUNAL:**

**Presidente:**

**Vocal:**

**Secretario:**

**FECHA DE DEFENSA:**

**CALIFICACIÓN:**

**PRESIDENTE VOCAL SECRETARIO**

Dedicado a mi familia y a todos aquellos……….

Declaración de Autoría

Yo, ...... con DNI ......, declaro que soy el único autor del trabajo fin de grado titulado “......” y que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual y que todo el material no original contenido en dicho trabajo está apropiadamente atribuido a sus legítimos autores.

Ciudad Real, a.....

Fdo: ......

Resumen

Esta plantilla de TFG está basada en los siguientes documentos:

* Plantilla de TFG de la Escuela Superior de Ingenieros Informáticos de Albacete.
* Plantilla de Latex desarrollada por Jesús Salido y disponible en el *sharepoint* de TFG
* Guía de estilo y formato para Trabajos Fin de Grado. Disponible en el *sharepoint* de TFG

Esta plantilla puede modificarse para adaptarse a las particularidades de cada Proyecto, tanto en contenido como en formato, siempre y cuando se respete las directrices básicas indicadas en la guía de estilo y formato arriba indicada.

Abstract

Agradecimientos

Índice general

[Capítulo 1 Introducción 19](#_Toc99974424)

[1.1. Motivación 19](#_Toc99974426)

[1.2. Redacción de la memoria 19](#_Toc99974427)

[1.3. Estructura del documento 20](#_Toc99974428)

[1.4. Abreviaturas y acrónimos 20](#_Toc99974429)

[Capítulo 2 Objetivos 21](#_Toc99974430)

[2.1. Objetivo general 21](#_Toc99974434)

[2.2. Objetivos específicos 22](#_Toc99974435)

[Capítulo 3 Metodología 23](#_Toc99974436)

[3.1. Guía rápida de las metodologías de desarrollo del software 23](#_Toc99974437)

[3.1.1 Proceso de desarrollo de software 23](#_Toc99974438)

[3.1.2 Metodologías de Desarrollo software 24](#_Toc99974439)

[3.2. Proceso de *testing* 25](#_Toc99974440)

[3.3. Marco tecnológico 25](#_Toc99974441)

[3.3.1 Herramientas CASE (Computer Aided Software Engineering) 26](#_Toc99974444)

[3.3.2 IDE (Integrated Development Environment) 26](#_Toc99974445)

[3.3.3 Depuración 26](#_Toc99974446)

[3.3.4 Repositorios y control de versiones 26](#_Toc99974447)

[3.3.5 Documentación 26](#_Toc99974448)

[3.3.6 Gestión y planificación de proyectos 26](#_Toc99974449)

[Capítulo 4 Resultados 27](#_Toc99974450)

[4.1. Resultados del TFG 27](#_Toc99974454)

[4.2. Listas y enumeraciones 27](#_Toc99974455)

[4.2.1 Listas 27](#_Toc99974459)

[4.2.2 Enumeraciones 27](#_Toc99974460)

[4.3. Tablas 28](#_Toc99974461)

[4.4. Figuras 28](#_Toc99974462)

[4.5. Código 29](#_Toc99974463)

[4.6. Bibliografía 29](#_Toc99974464)

[Capítulo 5 Conclusiones 30](#_Toc99974465)

[5.1. Revisión de los objetivos 30](#_Toc99974468)

[5.2. Presupuesto 30](#_Toc99974469)

[5.3. Competencias específicas de intensificación adquiridas y/o reforzadas 30](#_Toc99974472)

[Bibliografía 31](#_Toc99974473)

[Anexo I. Título del anexo 34](#_Toc99974474)

[I.1 Sección 1 del Anexo I 34](#_Toc99974475)

[I.2 Sección 2 del Anexo I 34](#_Toc99974476)

Índice de figuras

[**Figura 4.1** Toda figura debe tener un pie descriptivo 27](#_Toc97714824)

Índice de tablas

[**Tabla 4.1.** El título de la tabla se ha de colocar siempre encima de la tabla. Utilizar el estilo **Titulo Tabla** para formatear el encabezado de la tabla. 28](#_Toc99980086)

[**Tabla 4.2** Diseño de tabla con fila totales. 28](#_Toc99980087)

# Capítulo 1 Introducción

## 

El capítulo de *Introducción* debe describir el problema que se pretende resolver con el desarrollo del Trabajo Fin de Grado (TFG). Debe dar respuesta al *qué* sin especificar *cómo* se va a realizar, para lo cual se usarán el resto de los capítulos del documento. El lector de este documento debe tener claro el alcance del proyecto habiendo leído únicamente el capítulo de *Introducción*.



## Motivación

Esta sección aborda la motivación del trabajo. Se trata de señalar la necesidad que lo origina, su actualidad y pertinencia. Puede incluir también un estado de la cuestión (o estado del arte) en la que se revisen estudios o desarrollos previos y en qué medida sirven de base al trabajo que se presenta.

En este capítulo debería introducirse el *contexto disciplinar y tecnológico* en el que se desarrolla el trabajo de modo que pueda entenderse con facilidad el ámbito y alcance del TFG. Puesto que un TFG no tiene que ser necesariamente un trabajo con aportes novedosos u originales, solo es necesario la inclusión de *estado del arte* cuando este contribuya a aclarar aspectos clave del TFG o se desee justificar la originalidad del trabajo realizado. Si la sección *estado del arte* es muy extensa, considera la opción de introducirla como un capítulo independiente.

## Redacción de la memoria

Durante la realización de la memoria del TFG es importante tener presente respetar la guía de estilo de la institución (disponible en: [Trabajo Fin de Grado - Inicio (sharepoint.com)](https://pruebasaluuclm.sharepoint.com/sites/esicr/tfg/SitePages/Inicio.aspx)). Por tanto, el empleo de plantillas para un sistema de procesamiento de textos (por ejemplo, Word o LaTeX) puede requerir su adaptación cuando la plantilla mencionada no haya sido suministrada en la institución a la que se dirige el trabajo.

Para redactar un trabajo académico de modo efectivo se deben tener presentes una serie de normas que ayuden a conseguir un resultado final que sea claro y de fácil lectura.

A la hora de redactar el texto se debe poner especial atención en no cometer plagio y respetar los derechos de propiedad intelectual. En particular merece gran atención la inclusión de gráficos e imágenes procedentes de Internet que no sean de elaboración propia. En este sentido se recomienda consultar el manual de la Universidad de Cantabria[[1]](#footnote-1) en el que se explica de modo conciso cómo incluir imágenes en un trabajo académico.

## Estructura del documento

Este capítulo suele incluir una sección que indica la estructura (capítulos y anexos) del documento y el contenido de cada una de las partes en que se divide. Por tanto, las secciones que suelen acompañar este capítulo son:

* *Motivación*. Responde a la pregunta sobre la necesidad o pertinencia del trabajo.
* *Objetivo*. Determina de modo claro el propósito del trabajo descrito que puede desglosarse en subobjetivos cuando el objetivo principal se puede descomponer en módulos o componentes. Es muy importante definir el objetivo de modo apropiado. El Capítulo *Objetivos* de esta guía explica cómo definir el objetivo.
* *Antecedentes* o *Contexto disciplinar/tecnológico*. También puede denominarse *Estado del Arte* cuando se trata de comentar trabajos relacionados que han abordado la cuestión u objetivo que se plantea.
* *Estructura del documento*.Resumen de los capítulos y anexos que integran el documento.

## Abreviaturas y acrónimos

Un TFG que utiliza muchas abreviaturas y acrónimos puede añadir esta sección dónde se muestra el conjunto de abreviaturas y acrónimos y su significado.

# Capítulo 2 Objetivos

Para hacer un planteamiento apropiado de los objetivos se recomienda utilizar la Guía para la elaboración de propuestas de TFG en la que se explica cómo definir correctamente los objetivos de un TFG.



## Objetivo general

Introduce y motiva la problemática (i.e ¿cuál es el problema que se plantea y por qué es interesante su resolución?).

Debe concretar y exponer detalladamente el problema a resolver, el entorno de trabajo, la situación y qué se pretende obtener. También puede contemplar las limitaciones y condicionantes a considerar para la resolución del problema (lenguaje de construcción, equipo físico, equipo lógico de base o de apoyo, etc.). Si se considera necesario, esta sección puede titularse *Objetivos del TFG e hipótesis de trabajo*. En este caso, se añadirán las hipótesis de trabajo que el/la estudiante pretende demostrar con su TFG.

Una de las tareas más complicadas al proponer un TFG es plantear su *Objetivo*. La dificultad deriva de la falta de consenso respecto de lo que se entiende por *objetivo* en un trabajo de esta naturaleza. En primer lugar, se debe distinguir entre dos tipos de objetivo:

* La *finalidad específica* del TFG que se plantea para resolver una problemática concreta aplicando los métodos y herramientas adquiridos durante la formación académica. Por ejemplo, *Desarrollo de una aplicación software para gestionar reservas hoteleras on-line*.
* El *propósito académico* que la realización de un TFG tiene en la formación de un graduado. Por ejemplo, la *adquisición de competencias específicas de la intensificación* cursada.

En el ámbito de la memoria del TFG se tiene que definir el primer tipo de objetivo, mientras que el segundo tipo es el que se añade en el Capítulo de *Conclusiones* y que justifica las competencias específicas de la intensificación alcanzadas y/o reforzadas con la realización del trabajo.

La categoría del objetivo planteado justifica modificaciones en la organización genérica de la memoria del trabajo. Así en el caso de estudios y validación de hipótesis el apartado de resultados y conclusiones debería incluir los resultados de experimentación y los comentarios de cómo dichos resultados validan o refutan la hipótesis planteada.

## Objetivos específicos

Generalmente, el objetivo general puede ser descompuesto en varios objetivos más específicos que se pretenden alcanzar. En esta sección se enumeran y describen cada uno de ellos.

Junto con la definición de estos objetivos se puede especificar los *requisitos* que debe satisfacer la solución aportada. Estos requisitos especifican *características* que debe poseer la solución y *restricciones* que acotan su alcance. En el caso de un trabajo cuyo objetivo es el desarrollo de un *artefacto* los requisitos pueden ser *funcionales* y *no funcionales*.

Al redactar el objetivo de un TFG se debe evitar confundir los medios con el fin. Así es habitual encontrarse con objetivos definidos en términos de las *acciones* (verbos) o *tareas* que será preciso realizar para llegar al verdadero objetivo. Sin embargo, a la hora de planificar el desarrollo del trabajo si es apropiado descomponer todo el trabajo en *hitos* y estos en *tareas* para facilitar dicha *planificación*.

La categoría del objetivo planteado justifica modificaciones en la organización genérica de la memoria del trabajo. Así en el caso de estudios y validación de hipótesis el apartado de resultados y conclusiones debería incluir los resultados de experimentación y los comentarios de cómo dichos resultados validan o refutan la hipótesis planteada.

# Capítulo 3 Metodología

En este apartado se deben indicar las metodologías empleadas para planificación y desarrollo del TFG, así como explicar de modo claro y conciso cómo se han aplicado dichas metodologías.

## Guía rápida de las metodologías de desarrollo del software

A continuación, se incluye una guía rápida que puede ser de gran utilidad en la elaboración de este capítulo.

### Proceso de desarrollo de software

El *proceso de desarrollo de software* se denomina también *ciclo de vida del desarrollo del software* (SDLC, *Software Development Life-Cycle*) y cubre las siguientes actividades:

* Obtención y análisis de requisitos (*requirements analysis*). Es la definición de la funcionalidad del software a desarrollar. Suele requerir entrevistas entre los ingenieros de software y el cliente para obtener el `qué' y `cómo'. Permite obtener una especificación funcional del software.
* Diseño (*SW design*). Consiste en la definición de la arquitectura, los componentes, las interfaces y otras características del sistema o sus componentes.
* Implementación (*SW construction and coding*). Es el proceso de codificación del software en un lenguaje de programación. Constituye la fase en que tiene lugar el desarrollo de software.
* Pruebas (*testing and verification*). Verificación del correcto funcionamiento del software para detectar fallos lo antes posible. Persigue la obtención de software de calidad. Consisten en pruebas de *caja negra* y *caja blanca*. Las primeras comprueban que la funcionalidad es la esperada y para ello se verifica que, ante un conjunto amplio de entradas, la salida es correcta. Con las segundas se comprueba la robustez del código sometiéndolo a pruebas cuya finalidad es provocar fallos de software. Esta fase también incorpora las pruebas *de integración* en las que se verifica la interoperabilidad del sistema con otros existentes.
* Documentación (*documentation*). Persigue facilitar la mejora continua del software y su mantenimiento.
* Despliegue (*deployment*). Consiste en la instalación del software en un entorno de producción y puesta en marcha para explotación. En ocasiones implica una fase de *entrenamiento* de los usuarios del software.
* Mantenimiento (*maintenance*). Su propósito es la resolución de problemas, mejora y adaptación del software en explotación.

### Metodologías de Desarrollo software

*Las metodologías son el modo en que las fases del proceso software se organizan e interaccionan para conseguir que dicho proceso sea reproducible y predecible para aumentar la productividad y la calidad del software.*

Una metodología es una colección de:

* Procedimientos: indican cómo hacer cada tarea y en qué momento,
* Herramientas: ayudas para la realización de cada tarea, y
* Ayudas documentales.

Cada metodología es apropiada para un tipo de proyecto dependiendo de sus características técnicas, organizativas y del equipo de trabajo. En los entornos empresariales es obligado, a veces, el uso de una metodología concreta (p. ej. para participar en concursos públicos). El estándar internacional ISO/IEC 12270 describe el método para seleccionar, implementar y monitorear el ciclo de vida del software.

Mientras que unas intentan sistematizar y formalizar las tareas de diseño, otras aplican técnicas de gestión de proyectos para dicha tarea. Las metodologías de desarrollo se pueden agrupar dentro de varios enfoques según se señala a continuación.

* Metodología de Análisis y Diseño de Sistemas Estructurados (SSADM, *Structured Systems Analysis and Design Methodology*). Es uno de los paradigmas más antiguos. En esta metodología se emplea un modelo de desarrollo en cascada (*waterfall*). Las fases de desarrollo tienen lugar de modo secuencial. Una fase comienza cuando termina la anterior. Es un método clásico poco flexible y adaptable a cambios en los requisitos. Hace hincapié en la planificación derivada de una exhaustiva definición y análisis de los requisitos. Son metodologías que no lidian bien con la flexibilidad requerida en los proyectos de desarrollo software. Derivan de los procesos en ingeniería tradicionales y están enfocadas a la reducción del riesgo. Emplea tres técnicas clave:
  + Modelado lógico de datos (*Logical Data Modelling*),
  + Modelado de flujo de datos (*Data Flow Modelling*), y
  + Modelado de Entidades y Eventos (*Entity EventModelling*).
* Metodología de Diseño Orientado a Objetos (OOD, Object-Oriented Design). Está muy ligado a la OOP (Programación Orientada a Objetos) en que se persigue la reutilización. A diferencia del anterior, en este paradigma los datos y los procesos se combinan en una única entidad denominada *objetos* (o clases). Esta orientación pretende que los sistemas sean más modulares para mejorar la eficiencia, calidad del análisis y el diseño. Emplea extensivamente el Lenguaje Unificado de Modelado (UML) para especificar, visualizar, construir y documentar los artefactos de los sistemas software y también el modelo de negocio. UML proporciona una serie diagramas de básicos para modelar un sistema:
  + Diagrama de Clases (*Class Diagram*). Muestra los objetos del sistema y sus relaciones.
  + Diagrama de Caso de Uso (*Use Case Diagram*). Plasma la funcionalidad del sistema y quién interacciona con él.
  + Diagrama de secuencia (*Sequence Diagram*). Muestra los eventos que se producen en el sistema y como este reacciona ante ellos.
  + Modelo de Datos (*Data Model*).
* Desarrollo Rápido de Aplicaciones (RAD, Rapid Application Developmnent). Su filosofía es sacrificar calidad a cambio de poner en producción el sistema rápidamente con la funcionalidad esencial. Los procesos de especificación, diseño e implementación son simultáneos. No se realiza una especificación detallada y se reduce la documentación de diseño. El sistema se diseña en una serie de pasos, los usuarios evalúan cada etapa en la que proponen cambios y nuevas mejoras. Las interfaces de usuario se desarrollan habitualmente mediante sistemas interactivos de desarrollo. En vez de seguir un modelo de desarrollo en cascada sigue un modelo en espiral (Boehm). La clave de este modelo es el desarrollo continuo que ayuda a minimizar los riesgos. Los desarrolladores deben definir las características de mayor prioridad. Este tipo de desarrollo se basa en la creación de prototipos y realimentación obtenida de los clientes para definir e implementar más características hasta alcanzar un sistema aceptable para despliegue.
* *Metodologías Ágiles*. *"[...] envuelven un enfoque para la toma de decisiones en los proyectos de software, que se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto. Así el trabajo es realizado mediante la colaboración de equipos auto-organizados y multidisciplinarios, inmersos en un proceso compartido de toma de decisiones a corto plazo. Cada iteración del ciclo de vida incluye: planificación, análisis de requisitos, diseño, codificación, pruebas y documentación. Teniendo gran importancia el concepto de "Finalizado" (Done), ya que el objetivo de cada iteración no es agregar toda la funcionalidad para justificar el lanzamiento del producto al mercado, sino incrementar el valor por medio de "software que funciona" (sin errores). Los métodos ágiles enfatizan las comunicaciones cara a cara en vez de la documentación. [...]"*}[[2]](#footnote-2)

## Proceso de *testing*

Se debe indicar qué tipo de pruebas se han realizado, por ejemplo las siguientes:

* Pruebas modulares (pruebas unitarias). Su propósito es hacer pruebas sobre un módulo tan pronto como sea posible. Las pruebas unitarias que comprueban el correcto funcionamiento de una unidad de código. Dicha unidad elemental de código consistiría en cada función o procedimiento, en el caso de programación estructurada y cada clase, para la programación orientada a objetos. Las características de una prueba unitaria de calidad son: automatizable (sin intervención manual), completa, reutilizable, independiente y profesional.
* Pruebas de integración. Pruebas de varios módulos en conjunto para comprobar su interoperabilidad.
* Pruebas de caja negra.
* *Beta testing*.
* Pruebas de sistema y aceptación.
* ….

## Marco tecnológico

En esta sección se enumeran las tecnologías y herramientas utilizadas en la elaboración del TFG. A continuación, se citan algunos ejemplos.



### Herramientas CASE (Computer Aided Software Engineering)

Las herramientas CASE están destinadas a facilitar una o varias de las tareas implicadas en el ciclo de vida del desarrollo de software. Se pueden dividir en la siguientes categorías:

* Modelado y análisis de negocio.
* Desarrollo. Facilitan las fases de diseño y construcción.
* Verificación y validación.
* Gestión de configuraciones.
* Métricas y medidas.
* Gestión de proyecto. Gestión de planes, asignación de tareas, planificación, etc.

### IDE (Integrated Development Environment)

* Notepad++: <https://notepad-plus-plus.org/>
* Visual Studio Code: <https://code.visualstudio.com/>
* Atom: <https://atom.io/>
* GNU Emacs: <https://www.gnu.org/s/emacs/>
* NetBeans: <https://netbeans.org/>
* Eclipse: <https://eclipse.org/>
* QtCreator: <https://www.qt.io/ide/>
* jEdit: http://www.jedit.org/

### Depuración

* GNU Debugger: <https://www.gnu.org/s/gdb/>

### Repositorios y control de versiones

* Git: <https://git-scm.com/>
* Mercurial: <https://www.mercurial-scm.org/>
* Github: <https://github.com/>
* Bitbucket: <https://bitbucket.org/>
* SourceTree: https://www.sourcetreeapp.com/

### Documentación

* LaTeX: <https://www.latex-project.org/>
* Markdown: <https://markdown.es/>
* Doxygen: <http://www.stack.nl/\%7Edimitri/doxygen/index.html>
* DocGen: <http://mtmacdonald.github.io/docgen/docs/index.html>
* Pandoc: <http://pandoc.org/>

### Gestión y planificación de proyectos

* Trello: <https://trello.com/>
* Jira: <https://es.atlassian.com/software/jira>
* Asana: [https://asana.com/](https://asana.com/%20Slack)
* Slack: <https://slack.com/>
* Basecamp: <https://basecamp.com/>
* Teamwork Proyects: <https://www.teamwork.com/project-management-software>
* Zoho Projects: <https://www.zoho.com/projects/>

# Capítulo 4 Resultados

En los que se describen cómo se ha aplicado el método de trabajo para el caso concreto del TFG, incluyendo aquellos elementos (modelos, diagramas, especificaciones, etc.) más importantes y relevantes que se quieran hacer notar.



## Resultados del TFG

Este apartado debe explicar cómo el empleo de la metodología permite satisfacer tanto el objetivo principal como los específicos planteados en el TFG así como los requisitos exigidos (según exposición en capítulo *Objetivos*.

## Listas y enumeraciones



### Listas

Usar los estilos **Lista con viñetas** según el indentado necesario:

* Primer item: utilizar Lista con Viñetas

Continuar la descripción de item: Utilizar Lista

* + Subitem: utilizar Lista con Viñetas 2

Continuar un subitem: utilizar Lista 2

* + Subitem
    - Sub-Subitem: utilizar Lista con Viñetas 3

Continuar un subitem: utilizar Lista 3

### Enumeraciones

Utilizar enumeraciones mediante los estilos **Lista con números**:

1. Enumeración: Lista con números
2. Enumeración 2: Lista con números 2
3. Enumeración: Lista con números 3

## Tablas

Se recomienda utilizar el mismo formato de tabla para todo el documento. En el documento se ofrecen dos diseños de tabla: Tabla con bandas (ver Tabla 4.1) y Tabla con fila totales (ver Tabla 4.2). La tabla siempre ha de aparecer referenciada en el texto utilizando Referencias Cruzadas. Esto permite que la numeración de las tablas se actualice automáticamente, por ejemplo, Tabla 4.1. Utilizar Referencias, Referencia cruzada, Tipo Tabla, Sólo Rótulo y Número para añadir la referencia a la tabla.

**Tabla 4.1.** El título de la tabla se ha de colocar siempre encima de la tabla. Utilizar el estilo **Titulo Tabla** para formatear el encabezado de la tabla.

|  |  |  |
| --- | --- | --- |
| Columna 1 | Columna 2 | Columna 3 |
| Dato 1 | Dato 4 | Dato 6 |
| Dato 2 | Dato 5 | Dato 6 |
|  |  |  |
| Dato n | Dato n+1 | Dato n+2 |

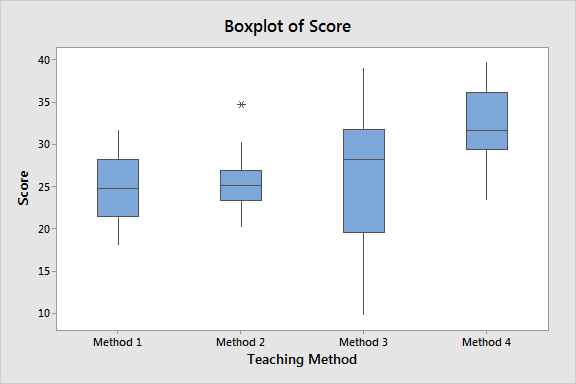
**Tabla 4.2** Diseño de tabla con fila totales.

|  |  |  |
| --- | --- | --- |
|  | Columna 1 | Columna 2 |
| Fila 1 | Dato 4 | Dato 6 |
| Fila 2 | Dato 5 | Dato 6 |
|  |  |  |
| Fila n | Dato n | Dato n+1 |
| Total | Total 1 | Total 2 |

## 

## Figuras

Al igual que ocurre con las tablas, toda figura tiene que estar referenciada en el texto, por ejemplo Figura 4.1. Utilizar Referencias, Referencia cruzada, Tipo Figura, Sólo Rótulo y Número. Seleccionar la imagen y aplicar el estilo Figura para ajustarla correctamente en el documento.



**Figura 4.1** Toda figura debe tener un pie descriptivo

## Código

Para describir el código o pseudo-código en el documento utilizar el estilo Código:

**def** search (initial)

open = [initial]

**while** open

node = open.pop()

**if** test (node)

**return** node

**return** failure

**Código 4.1.** Código search

## Bibliografía

Se recomienda la utilización de un gestor de referencias como Mendeley o Endnote. Toda referencia que aparezca en la Sección de Bibliografía ha de aparecer referenciada en el texto. Un ejemplo de referencia en el texto sería (Babar, Kitchenham, Zhu, Gorton, & Jeffery, 2006).

Además, se recomienda utilizar estilos estándar para la bibliografía, por ejemplo, el de American Psycological Association (APA) 6th Edition.

# Capítulo 5 Conclusiones

En este capítulo se debe incluir el juicio crítico y discusión sobre los resultados obtenidos. Si es pertinente deberá incluir información sobre trabajos derivados como publicaciones o ponencias, así como trabajos futuros, solo si estos están planificados en el momento en que se redacta el texto. Incluirá obligatoriamente la justificación de las competencias de la tecnología específica cursada por el estudiante que se han adquirido durante el desarrollo del TFG



## Revisión de los objetivos

En esta sección se deberá revisar en qué grado se han completado los objetivos fijados al principio del proyecto. Se deberá también indicar las posibles desviaciones de los objetivos fijados, así como de la planificación, y tratar de justificar tales desviaciones.

## Presupuesto

Si el TFG consiste en el desarrollo e implementación de un prototipo, la memoria debe incluir el coste del prototipo considerando tanto el hardware como los recursos humanos necesarios para su desarrollo.

Cuando se tiene en cuenta la puesta en marcha de un proyecto de ingeniería, la planificación y presupuesto que se realizan de modo previo a su ejecución son críticos para gestionar los recursos que permitan alcanzar los objetivos de calidad, temporales y económicos previstos para el proyecto.

Es muy importante que todas las justificaciones aportadas se sustenten no solo en juicios de valor sino en evidencias tangibles como: historiales de actividad, repositorios de código y documentación, porciones de código, trazas de ejecución, capturas de pantalla, demos, etc.



## Competencias específicas de intensificación adquiridas y/o reforzadas

Se deberán listar aquellas competencias de la intensificación que hayan sido adquiridas y/o reforzadas con el desarrollo de este TFG, incluyendo su justificación.

Bibliografía

(Estos ejemplos siguen el formato American Psycological Association (APA) 6th Edition: [Estilo APA 7ª ed. - Citas y elaboración de bibliografía: el plagio y el uso ético de la información - Biblioguías at Universidad Autónoma de Madrid (uam.es)](https://biblioguias.uam.es/citar/estilo_apa_7th_ed))

Alexa Internet Inc. (2014). The top 500 sites on the web. Retrieved February 11, 2014, from http://www.alexa.com/topsites

Anda, B., Sjøberg, D., & Jørgensen, M. (2001). Quality and Understandability of Use Case Models. In 15th European Conference on Object-Oriented Programming (ECOOP’01) (pp. 402–428). London, UK: Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45337-7\_21

Babar, M. A., Kitchenham, B. A., Zhu, L., Gorton, I., & Jeffery, R. (2006). An empirical study of groupware support for distributed software architecture evaluation process. Journal of Systems and Software, 79(7), 912–925. https://doi.org/10.1016/j.jss.2005.06.043

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). The Goal Question Metric Approach. In Encyclopedia of Software Engineering (Vol. 2, pp. 528–532). Wiley. Retrieved from http://wwwagse-old.informatik.uni-kl.de/pubs/repository/basili94b/encyclo.gqm.pdf

Basili, V. R., Shull, F., & Lanubile, F. (1999). Building Knowledge through Families of Experiments. IEEE Transactions on Software Engineering, 25(4), 456–473. https://doi.org/10.1023/A:1009742216007

Bereiter, C. (2002). Education and Mind in the Knowledge Age (1st ed.). Routledge. Retrieved from http://www.amazon.com/Education-Mind-Knowledge-Carl-Bereiter/dp/0805839437

Biostat Inc. (2006). Comprehensive Meta-Analysis. Retrieved April 17, 2013, from http://www.meta-analysis.com/

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). The Unified Modeling Language User Guide (2nd ed.). Addison-Wesley Professional.

Canfora, G., Cimitile, A., Garcia, F., Piattini, M., & Visaggio, C. A. (2006). Evaluating advantages of test driven development. In 2006 ACM/IEEE international symposium on International symposium on empirical software engineering (ISESE’06) (p. 364). Rio de Janeiro, Brazil: ACM Press. https://doi.org/10.1145/1159733.1159788

Castro, J., Kolp, M., & Mylopoulos, J. (2001). A requirements-driven development methodology. In 13th Int. Conf. On Advanced Information Systems Engineering (CAiSE’01) (pp. 108–123). London, UK: Springer-Verlag. https://doi.org/10.1007/3-540-45341-5\_8

Celko, J., Davis, J. S., & Mitchell, J. (1983). A demonstration of three requirements language systems. ACM SIGPLAN Notices, 18(1), 9–14. https://doi.org/10.1145/948093.948094

Cockburn, A. (2000). Writing Effective Use Cases (1st ed.). Addison-Wesley Professional.

Cruz-Lemus, J. A., Genero, M., Caivano, D., Abrahão, S., Insfrán, E., & Carsí, J. A. (2011). Assessing the influence of stereotypes on the comprehension of UML sequence diagrams: A family of experiments. Information and Software Technology, 53(12), 1391–1403. https://doi.org/10.1016/j.infsof.2011.07.002

Cruz-Lemus, J. A., Genero, M., Manso, M. E., Morasca, S., & Piattini, M. (2009). Assessing the understandability of UML statechart diagrams with composite states—A family of empirical studies. Empirical Software Engineering, 14(6), 685–719. https://doi.org/10.1007/s10664-009-9106-z

Cruz-Lemus, J. A., Maes, A., Genero, M., Poels, G., & Piattini, M. (2010). The impact of structural complexity on the understandability of UML statechart diagrams. Information Sciences, 180(11), 2209–2220. <https://doi.org/10.1016/j.ins.2010.01.026>.

Cysneiros, L. M., & Yu, E. S.-K. (2004). Non-Functional Requirements Elicitation. In J. C. S. do Prado Leite & J. H. Doorn (Eds.), Perspectives on Software Requirements (pp. 115–138). Springer US. https://doi.org/10.1007/978-1-4615-0465-8\_6

Damian, D. (2001). An empirical study of requirements engineering in distributed software projects: is distance negotiation more effective? In 8th Asia-Pacific Software Engineering Conference (APSEC’01) (pp. 149–152). Macao, China: IEEE Comput. Soc. https://doi.org/10.1109/APSEC.2001.991471

De Lucia, A., Fasano, F., Oliveto, R., & Tortora, G. (2006). Can Information Retrieval Techniques Effectively Support Traceability Link Recovery? In 14th IEEE International Conference on Program Comprehension (ICPC’06) (pp. 307–316). Athens, Greece: IEEE. https://doi.org/10.1109/ICPC.2006.15

Dieste, O., Fernández, E., García Martínez, R., & Juristo, N. (2011). Comparative analysis of meta-analysis methods: when to use which? In 15th International Conference on Evaluation & Assessment in Software Engineering (EASE’11) (pp. 36–45). Durham, UK: IET.

Dourish, P., & Bellotti, V. (1992). Awareness and coordination in shared workspaces. In ACM conference on Computer-supported cooperative work (CSCW’92) (pp. 107–114). Toronto, Canada: ACM Press. https://doi.org/10.1145/143457.143468

1. Título del anexo
   1. Sección 1 del Anexo I

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

* 1. Sección 2 del Anexo I

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?.

1. Guía de Imágenes: https://web.unican.es/buc/Documents/Formacion/guia\_imagenes.pdf [↑](#footnote-ref-1)
2. Fuente: Wikipedia [↑](#footnote-ref-2)