DOCUMENTACIÓN

JAVIER CÁMARA MORENO

PRUEBA DE ACCESO AL CUERPO DE PROFESOR TITULAR DE UNIVERSIDAD
DEL ÁREA DE LENGUAJES Y SISTEMAS INFORMÁTICOS

Málaga, 27 de enero de 2022

Preámbulo

Por resolución de 9 de noviembre de 2021, BOE núm. 277 de 19 de noviembre de 2021, la Universidad de Málaga convocó a concurso de acceso, entre otras, la plaza de Profesor Titular de Universidad cuyos detalles se muestran a continuación. El presente documento se ha elaborado para cumplir parcialmente con las instrucciones establecidas en dicha convocatoria en relación a la documentación a entregar en el acto de presentación del concurso, a saber, y en este orden:

"*Historial académico, docente e investigador y, en su caso, asistencial sanitario, por sextuplicado, en el que se detallarán los méritos hasta la fecha de presentación de solicitudes, según modelo normalizado de currículum que utiliza la Agencia Nacional de Evaluación de la Calidad y la Acreditación para la acreditación nacional para el acceso al cuerpo de Profesores Titulares de Universidad, que se adjunta como Anexo V a la presente resolución.*

Proyecto investigador, por sextuplicado, que pretenda desarrollar el candidato o candidata, conforme a la actividad docente e investigadora que conste en la convocatoria de la plaza.

Proyecto docente, por sextuplicado, referido a una asignatura obligatoria adscrita al área de conocimiento de la plaza objeto de concurso e incluida en el correspondiente plan de estudios de una titulación oficial de Grado o Máster de la Universidad de Málaga, con validez en todo el territorio nacional y que esté vigente en la fecha de publicación de la convocatoria de la plaza en el BOE.

Un resumen, por sextuplicado, del tema elegido previamente por el candidato o candidata de entre los presentados en el proyecto docente para su exposición oral. El resumen no podrá exceder de 25 páginas."

Cuerpo	Profesor Titular de Universidad
Número de plaza	037TUN21
Área de conocimiento	Lenguajes y Sistemas Informáticos
Departamento	Lenguajes y Ciencias de la Computación
Perfil docente	Docencia en Introducción a la Ingeniería del Software, asignatura adscrita al Área de Conocimiento
Perfil investigador	Investigación en software system structures y software functional properties.

A fin de facilitar la labor del tribunal, se proporciona copia digital de toda la documentación, incluida la documentación acreditativa de los méritos alegados en el curriculum vitae, disponible en el siguiente enlace:

https://javier-camara.github.io/files/tun37.html

PROYECTO INVESTIGADOR

JAVIER CÁMARA MORENO

PRUEBA DE ACCESO AL CUERPO DE PROFESOR TITULAR DE UNIVERSIDAD
DEL ÁREA DE LENGUAJES Y SISTEMAS INFORMÁTICOS

Málaga, 27 de enero de 2022

Preámbulo

Por resolución de 9 de noviembre de 2021, BOE núm. 277 de 19 de noviembre de 2021, la Universidad de Málaga convocó a concurso de acceso, entre otras, la plaza de Profesor Titular de Universidad cuyos detalles se muestran a continuación. El presente documento se ha elaborado para cumplir parcialmente con las instrucciones establecidas en dicha convocatoria en relación a la documentación a entregar en el acto de presentación del concurso, a saber:

"*Proyecto investigador, por sextuplicado, que pretenda desarrollar el candidato o candidata, conforme a la actividad docente e investigadora que conste en la convocatoria de la plaza."*

Cuerpo	Profesor Titular de Universidad
Número de plaza	037TUN21
Área de conocimiento	Lenguajes y Sistemas Informáticos
Departamento	Lenguajes y Ciencias de la Computación
Perfil docente	Docencia en Introducción a la Ingeniería del Software, asignatura adscrita al Área de Conocimiento
Perfil investigador	Investigación en s*oftware system structures* y *software functional properties*.

ÍNDICE

1. Experiencia en el perfil investigador de la plaza

El perfil investigador de la plaza se centra en la investigación en *software system structures* y *en software functional properties*. Ambos términos se han extraído de la *ACM Computing Classification System*[7] propuesta en 2012.

El primer término, *Software system structures*, se encuentra anidado en la categoría *Software and its engineering -> Software organization and properties -> Software system structures*. Este término, por tanto, se encuentra directamente relacionado con el diseño y la organización de sistemas software en distintos ámbitos. En el siguiente nivel de anidamiento, bajo Software system structures, encontramos términos tales como *Software architectures*, *Software system models* y *Abstraction, modeling and modularity*. Todos estos términos tienen relación con la experiencia investigadora del candidato y con el proyecto de investigación que se presenta en este documento.

Respecto al segundo término del perfil investigador, Software funcional properties, éste se encuentra anidado en la categoría *Software and its engineering -> Software organization and properties -> Software functional properties.* En el siguiente nivel de anidamiento, bajo *Software functional properties*, encontramos los términos *Correctness* y *Formal methods*. Por tanto, el término *Software functional properties* está relacionado con el uso de técnicas de verificación que son de utilidad para obtener garantías formales sobre la *corrección* del software.

El solicitante tiene una amplia experiencia investigadora tanto en el desarrollo de técnicas de modelado, análisis y síntesis de *arquitecturas software*, así como en la aplicación de *métodos formales* para verificar la corrección de sistemas software. Concretamente, una gran parte de su investigación durante los últimos años se ha centrado en el desarrollo y utilización de técnicas formales de verificación focalizada en los llamados *quantitative aspects of correctness* [1][2], que abarcan el análisis de propiedades funcionales y no funcionales en *software-intensive systems* tales como sistemas autónomos y adaptativos (*self-adaptive*), sistemas IT complejos, y sistemas ciberfísicos, entre otros.

En cuanto a las publicaciones que respaldan su actividad investigadora, caben destacar los artículos de revista [3]–[11], en los que el solicitante ha trabajado fundamentalmente en dos líneas de investigación. Por un lado, la de la provisión de garantías formales en *self-adaptive*

[7] https://www.acm.org/publications/class-2012

systems, en la que se ha utilizado fundamentalmente técnicas de análisis basadas en *probabilistic model checking* [12] para dar garantías cuantitativas formales bajo incertidumbre [4]–[6], [8]–[11], combinándolas en algunos casos con técnicas de testing [6], [10]. Además de las anteriores publicaciones en revista, esta línea de investigación ha sido avalada por un *ACM SigSoft Distinguished Paper Award* en la conferencia *Quality of Software Architectures* (QoSA) de 2014 [13], paper que originó la publicación en revista [6]. Por otro lado, otra línea ha explorado el uso combinado de *lightweight formal methods* como *Alloy* [14] con *probabilistic model checking* para la síntesis de espacios de diseño arquitectónico con garantías estructurales, de comportamiento y cuantitativas [3], que se ha generalizado recientemente en una herramienta que permite tratar con otros tipos de modelos estructurales no arquitectónicos [15]. Esta última línea de investigación fue respaldada con el *Best Paper Award* en la *European Conference on Software Architecture* de 2017 [16], artículo que dio lugar a la publicación de revista [3].

2. Experiencia en proyectos, propuestas de investigación y liderazgo

2.1 . Participación en proyectos de investigación

Como aparece reflejado en su CV, el candidato ha participado en 11 proyectos internacionales (3 Reino Unido, 6 en Estados Unidos, 1 Portugal, 1 Francia), dos nacionales del Plan Nacional de I+D+i, y uno regional (Junta de Andalucía). En el ámbito internacional, el candidato ha participado como investigador en seis proyectos estadounidenses financiados por la *Defense Advanced Research Projects Agency* (DARPA), *la National Security Agency* (NSA), la *Office of Naval Research* de la *US Navy* (ONR), el *US Department of Defense* (DoD), y la *National Aeronautics and Space Administration* (NASA). En Portugal y Francia, los proyectos en los que participó el candidato fueron financiados por la *Fundação para a Ciência e a Tecnologia* (FCT) y la *Agence Nationale de la Recherche* (ANR), respectivamente. En el Reino Unido, el candidato ha participado en proyectos financiados por el *Defence Science and Technology Laboratory* (Dstl) del *Ministry of Defence* (MoD), la *UK Atomic Energy Authority* (UKAEA), y el *Assuring Autonomy International Programme* (AAIP).

Entre todos estos proyectos, cabe destacar la participación del candidato como Co-IP en los proyectos *Assured Mission-Critical Applications for Teams of Unmanned* Vehicles (AMCA, financiado por Dstl), *Robotic Assistive Care* (ALMI, financiado por AAIP), y *Multi-Robot Systems for the Inspection and Maintenance of Nuclear Fusion Infrastructure* (financiado por la autoridad de la energía nuclear del Reino Unido UKAEA), todos ellos directamente relacionados con el perfil investigador de la plaza mediante la utilización de *métodos formales* para la obtención de garantías sobre la *corrección* del software de los sistemas construidos.

2.2 . Participación en propuestas de investigación

El candidato ha sido profesor (*Lecturer*) en la *University of York* en Reino Unido (2018-2021). Durante este periodo, el candidato participó en la redacción de varias propuestas de proyectos de investigación que fueron financiadas con éxito, y que incluyen:

- AMCA: Assured Mission-Critical Applications for Teams of Unmanned Vehicles, Dstl, £90,768.00 (como Co-IP)
- Robotic Assistive Care, LLOYD'S REGISTER FOUNDATION (a través de AAIP), £232,568.80 (como Co-IP)
- Assured and Scalable Self-Adaptation for the Engineering of Trustworthy Autonomous Robotic Teams. Dstl, £99,947.22 (como Co-IP, financiada pero finalmente cancelada por falta de candidatos adecuados para formación del equipo técnico debido a restricciones de nacionalidad).
- RASPBERRY SI: Resource Adaptive Software Purpose-Built for Extraordinary Robotic Research Yields - Science Instruments. NASA, $249,975.00 (Como investigador colaborador)
- Multi-Robot Systems for the Inspection and Maintenance of Nuclear Fusion Infrastructure, RACE-UKRI, £68,000.00 (Como Co-IP)
- Assurance of Online Learning for Robotic and Autonomous Systems. LLOYD'S REGISTER FOUNDATION (a través de AAIP), £113,577 (Como Co-IP, comienzo en Enero de 2022)

Además de las propuestas anteriores, el candidato también envió otra propuesta al *Engineering and Physics Research Council* (organismo de ámbito nacional perteneciente al gobierno que financia los proyectos en ciencia e ingeniería en el Reino Unido) como Co-IP junto a otro equipo del *Imperial College* que no fue financiada finalmente. El candidato también estuvo contratado como investigador postdoctoral (2013-2015) y *Systems Scientist* (posición permanente centrada en investigación, 2015-2018) en la *Carnegie Mellon University* (Pittsburgh, Estados Unidos). Durante este periodo, el candidato participó en la redacción de varias propuestas, una de ellas como Co-IP (que no fue finalmente considerada para su financiación) para la *National Science Foundation* (NSF).

2.3 Liderazgo

El candidato ha dirigido, junto con el profesor Radu Calinescu, la tesis por la Universidad de York del doctorando Saud Yonbawi (ahora profesor ayudante en la Universidad de Jeddah, en

Arabia Saudí). La tesis fue defendida con éxito en Junio de 2021. La tesis versa sobre el control descentralizado en sistemas adaptativos distribuidos que requieren garantías formales estrictas de calidad de servicio.

Además, el candidato dirige actualmente (también junto al profesor Radu Calinescu), otras dos tesis doctorales por la Universidad de York. La primera, correspondiente al doctorando Brendan Devlin-Hill se centra en la provisión de garantías formales en sistemas multi-robot heterogéneos que operan en entornos extremos (en concreto, en reactores de fusión nuclear como los planeados en el proyecto *Iter* en el que colaboran más de 35 países http://www.iter.org). La segunda, del doctorando Hamish Zhang, versa sobre la provisión de garantías formales en sistemas autónomos y adaptativos que incorporan aprendizaje automático *online*, es decir, que ocurre de forma continua en tiempo de ejecución.

Además de la supervisión de doctorandos, el candidato supervisa (también junto a Radu Calinescu) a Jordan Hamilton, un investigador postdoctoral en el contexto del proyecto ALMI.

Anteriormente a su etapa en York, el candidato también fue supervisor asistente en dos tesis doctorales por la *Carnegie Mellon University* dirigidas por el profesor David Garlan. La primera, del doctorando Gabriel Moreno (ahora *Principal Investigator* en el *CMU Software Engineering Institute*), versa sobre la consideración explícita del tiempo de latencia en sistemas adaptativos y fue defendida con éxito en 2017. La segunda, del doctorando Ashutosh Pandey (ahora *Research Scientist* en Facebook), versa sobre la planificación bajo incertidumbre en sistemas adaptativos y fue defendida con éxito en 2019.

3 Modelos de financiación

La financiación es el pilar básico de la investigación en la universidad pública. Sin financiación, no se puede conformar un equipo que lleve a cabo tareas de investigación. La financiación para proyectos tecnológicos proviene de dos fuentes principales: los gobiernos y las empresas. El personal investigador participa en convocatorias para la financiación de sus proyectos. Estas subvenciones requieren un proceso de selección, en el que las agencias consideran el perfil y la trayectoria de los investigadores, las instalaciones y equipamiento necesario, el tiempo involucrado y el potencial de los resultados de la producción científica.

A continuación, se muestra un listado, no exhaustivo, de algunos de los medios de financiación pública en el sector tecnológico en las universidades españolas:

- **Proyectos Europeos.** El próximo programa de inversión en investigación e innovación de la Unión Europea se llama *Horizonte Europa*. Servirá para financiar proyectos entre 2021 y 2027. Los proyectos financiados por la Unión Europea se proponen por

consorcios de empresas e instituciones, tales como los pasados Proyectos Europeos FP7. Mediante proyectos de este tipo es posible contratar investigadores postdoctorales, predoctorales y técnicos. Es común que se realicen tesis doctorales en el contexto de un Proyecto Europeo.

- **Proyectos Nacionales.** Se trata de los proyectos financiados por el Gobierno de España, que suelen ser una vía de financiación común de los grupos de investigación españoles. Estos proyectos ofrecen financiación durante 3 o 4 años y están liderados por uno o dos IP con una trayectoria investigadora consolidada. Suelen permitir la contratación de personal técnico y también son una vía de acceso a la universidad, pues algunos de estos proyectos llevan asociada la contratación de personal investigador para la realización de la tesis doctoral.
- **Proyectos Regionales.** Dependen de cada comunidad autónoma. En Andalucía tenemos los proyectos financiados por la Junta de Andalucía. Su objetivo es similar al de los proyectos nacionales, y en Andalucía es común que distintos grupos de investigación gocen a la vez de proyectos nacionales y regionales. Los proyectos de la Junta de Andalucía también permiten la contratación de personal técnico y también son una vía de acceso a la universidad.
- **Redes Nacionales.** Estas redes, financiadas por el Gobierno de España, están destinadas a la colaboración entre grupos de investigación españoles. La financiación recibida es pequeña, y está principalmente destinada a cubrir los gastos de desplazamiento de personal investigador entre distintos centros españoles.
- **Financiación para investigadores.** La financiación descrita en los puntos anteriores está destinada a grupos de investigación. Sin embargo, también hay convocatorias y ayudas destinadas a financiar a un solo investigador, y suelen estar encaminadas a la movilidad internacional e intersectorial y la retención de talento. Algunas de estas convocatorias y ayudas requieren la presentación de un proyecto investigador, mientras que otras no lo requieren. Por ejemplo, La Unión Europea propone las acciones *Marie Sklodowska-Curie* (antes llamadas becas *Marie Curie*), que ofrecen ayudas tanto a investigadores predoctorales como posdoctorales y cuyo objetivo es proporcionar a los investigadores una formación basada en la excelencia y con las mejores oportunidades de desarrollo. La Unión Europea también propone ayudas destinadas a un único investigador, con una trayectoria excelente, para que éste forme un equipo de investigación, tales como la *Starting Grant* o la *Consolidator Grant.* Estas ayudas permiten al solicitante conformar, desde cero, un grupo de investigación sólido, y para optar a ellas hay que proponer un proyecto investigador excelente. El Gobierno de España propone convocatorias de proyectos nacionales para un único solicitante, los llamados Proyectos JIN para jóvenes investigadores sin vinculación contractual o con vinculación temporal limitada. También se convocan anualmente becas predoctorales,

las conocidas ayudas para la formación de profesorado universitario (FPU), y becas postdoctorales, como las ayudas para contratos Juan de la Cierva (formación e incorporación) y las ayudas para contratos Ramón y Cajal.

Además de las ayudas y convocatorias descritas, hay muchas otras que intentan acercar el mundo universitario y el empresarial. Ejemplos son:

- **Ayudas para contratos Torres Quevedo**, cuyo objetivo es la contratación laboral de personas con el grado de doctor para promover la realización de proyectos de investigación industrial, de desarrollo experimental o estudios de viabilidad previo.
- **Ayudas para la formación de doctores en empresas "Doctores Industriales".** Su objetivo es la formación de doctores en empresas mediante la cofinanciación de los contratos laborales del personal investigador en formación que participen en un proyecto de investigación industrial o de desarrollo experimental que se desarrolle en la empresa, en el que se enmarcará su tesis doctoral.
- **Convocatoria Retos-Colaboración**. Está destinada a financiar proyectos de colaboración entre empresas y centros de investigación públicos y privados para promover la investigación orientada a la resolución de los retos de la sociedad especificados en el Plan estatal de I+D+i mediante el desarrollo de nuevos productos, procesos y servicios.

4 Proyecto investigador propuesto

El proyecto investigador presentado en la segunda parte de este documento se podría enmarcar en diferentes convocatorias de las detalladas en el punto anterior. Se presenta en el contexto del grupo de investigación SCENiC (http://www.scenic.uma.es), perteneciente al Grupo de Ingeniería del Software (GISUM) del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga y componente del Instituto de Tecnología e Ingeniería del Software (ITIS) de la misma universidad.

El proyecto trata sobre la construcción sistemática de herramientas de análisis capaces de proveer garantías formales en sistemas software sujetos a incertidumbre (procedente de la interacción con personas, interacciones complejas con elementos físicos en sistemas ciberfísicos, variabilidad de recursos disponibles, etc.). Estas herramientas permitirán incorporar elementos cuantitativos, de comportamiento, y estructurales en las garantías provistas, que normalmente son analizados de forma independiente con distintas técnicas y herramientas (p.ej: *probabilistic model checkers*, *constraint solvers*). Los métodos y herramientas desarrollados por el proyecto emplearán técnicas de ingeniería del software basada en modelos (MBSE) para facilitar la interoperabilidad entre las distintas herramientas

y garantizar la consistencia de la información contenida en las especificaciones formales utilizadas en cada una de ellas. Para mejorar la escalabilidad del análisis, también se utilizarán técnicas de ingeniería del software basada en búsqueda (SBSE) que permitan a las herramientas construidas acelerar la convergencia hacia soluciones cuasi-óptimas con garantías formales (p.ej: para la síntesis de controladores o configuraciones arquitectónicas con garantías probabilísticas de QoS y de satisfacción de restricciones de comportamiento y estructurales del dominio de aplicación).

La estructura y orientación del proyecto están alineadas con las convocatorias para "proyectos de generación de conocimiento" en modalidad individual publicadas por el Ministerio de Ciencia e Innovación. Dado que uno de los requisitos obligatorios para estas convocatorias es la presentación de la memoria científico-técnica de la propuesta de proyecto en inglés si se solicitan 100.000 EUR o más en costes directos, la memoria que se presenta a continuación está en consecuencia escrita en inglés al ser el coste de la propuesta realizada superior al límite establecido para que ese requisito sea obligatorio.

Este proyecto ha sido adaptado para ser utilizado como núcleo técnico de una propuesta de proyecto de Transición Ecológica y Digital (TED) de la convocatoria 2021 realizada por el Ministerio de Ciencia e Innovación, en la que la tecnología propuesta se pone al servicio de la sostenibilidad urbana. El proyecto ha sido presentado en enero de 2022 y está pendiente de evaluación.

1. Context and Research Problem

Formal methods used to validate software designs like Alloy[14], OCL [17], Z [18], B [19], and VDM [20] share a common conceptual foundation that enables designers to describe rich structures (e.g., software architectures [21], [22], database object-relational mappings [23], network and security models [24], [25]) as sets of relational constraints and reason about them systematically. Among these methods, Alloy has been particularly successful in a broad range of applications thanks to its lightweight approach to formal specification and tool support to explore collections of structures implicitly specified by a set of constraints written in its relational logic.

Despite their advantages, these methods are not equipped for analyzing systems subject to objective (or aleatoric) *uncertainties*, which commonly affect modern software-intensive systems and are derived, for instance, from the lack of control over third-party system components (e.g., residing in the cloud), humans in the loop, and complex interactions between software and physical elements in cyber-physical systems [26].

During the last two decades, a different class of formal technique, commonly referred to as *quantitative verification* or *probabilistic model checking* [2], [12], [27], has emerged as a powerful way to reason under uncertainty (characterized as stochastic behavior) about quantitative aspects of software systems like performance [28]–[30], reliability [31]–[33], and security [34]–[36]. However, modern probabilistic model checkers like PRISM [37] and STORM [38] employ notations like the PRISM language, the PEPA process algebra [39], and the conditional probabilistic guarded command language cpGCL [40] that do not retain the flexibility in describing structures of relational methods like Alloy. Concretely, models written in these languages have a fixed structure, allow variation only in quantitative parameters, and are not equipped with any guarantees about their structural consistency (e.g., satisfaction of architectural and domain constraints).

Structure, stochastic behavior, and their interplay can distinctly impact functional and nonfunctional requirement satisfaction in systems subject to uncertainty [41]–[43]. *Unfortunately, the lack of methods combining the strengths of relational modeling and quantitative verification forces designers to trade systematic exploration of sound alternative structural designs for analytic capabilities that concern probabilistic and other quantitative guarantees on system properties. This hampers the analysis of formal guarantees under uncertainty across design spaces, (i.e., in the presence of structural variability).*

To mitigate this situation and work towards the *provision of formal assurance evidence* in systems that are subject to uncertainty and structural changes (e.g., smart cities, microservice-based architectures, as well as autonomous, self-adaptive, and socio-cyber-physical systems in general), our prior exploratory research proposed a method to enhance structural modeling and synthesis (in Alloy) with quantitative guarantees in the style provided by probabilistic model checkers and quantitative verification (which we elaborate in the next section), where the focus was put on language design (both for system modeling and for capturing system properties in extended temporal logics) [15]. We validated our methods by building a prototype tool that produced formal specifications to be used as input by the Alloy analyzer and the probabilistic model checker PRISM, which were used in tandem and carefully coordinated.

Despite its promising results, our experience revealed that putting our vision into practice poses significant challenges that remain to be addressed. Concretely, developing the analytical mechanisms required to provide tool support for our class of approach demands integrating, in a harmonious way, disparate formalisms (that capture structure, stochastic behavior –including changes in system structure–, and quantitative properties), as well as reasoning mechanisms like constraint solving (for structural synthesis) and iterative numerical methods (for probabilistic model checking). This makes (C1) guaranteeing the *consistency* across models produced in the different formalisms a key issue to preserve the formal guarantees provided by such an approach. Secondly, there is a challenge (C2) associated with the *generalization* of the approach, to allow integrating new formalisms and types of analysis (e.g., stochastic games for worst-case scenario analysis), as well as tools (e.g., SMT solvers like Z3 [44] and statistical model checkers) to eliminate major dependencies from specific tools. Thirdly, equipping these modeling languages and analysis mechanisms to (C3) capture and analyze *dynamic behavior* in which the structure of the system changes (e.g., robot software or microservice architecture that reconfigures itself at run time), as opposed to a set of alternative fixed structural configurations demands careful consideration of the language constructs required to capture such behaviors and their properties, as well as adequate coordination of different analysis techniques and tools. Furthermore, this complex analytical machinery must (C4) *scale* to enable property checking in realistic problem spaces.

In this project, we propose to address these challenges at a foundational level by devising methods and tools to automate joint reasoning about the structure and stochastic behavior of systems (including those that entail structural changes). In order to achieve our goal, we will employ: (i) Model-Based Software Engineering (MBSE) techniques to automate the translation of domain models into various formal specifications, preserving the consistency among them, facilitating maintainability and integration of new formalisms and tools, and (ii) Search-Based Software Engineering (SBSE) techniques to scale up the analytical capabilities of formal

techniques to real scenarios. We elaborate how to address these research challenges in our approach in the following sections.

2. Supporting Concepts and Technology

2.1 Lightweight Formal Methods and Relational Modeling

Pioneering formal methods like Z [18] have expressive notations that limit the level of automation of theorem proving, which requires guidance from an experienced user. In contrast, Alloy [14] can be considered a first-order subset of Z that bounds models to a finite scope, and this makes it automatically analyzable. Other techniques and languages like VDM [20] and OCL [17] are based on first-order logic and include tools that can support design-time analysis that allows exhaustive search over a finite space of cases, analogously to Alloy. These methods have a strong focus on enabling modeling of rich structures, but are limited in terms of reasoning about complex concurrent behaviors. Other methods with similar formal underpinnings like Event- B [45] include more sophisticated constructs to reason efficiently about concurrent behaviors on top of structures. Concretely, Event-B specifications consist of static and dynamic aspects (called contexts and machines, respectively). Contexts define types (sets) and give information about them in axioms. Analogously to signatures in Alloy, contexts can extend other contexts to use their information and augment them.

The notion of finite models and type hierarchy incorporated in Alloy and Event-B, along with the ability to capture complex relational constraints, endows these methods with a high level of flexibility to: (i) capture multiple topological design variants implicitly in a single specification (i.e., those that satisfy the specification's relational constraints), and (ii) automatically synthesize explicit descriptions of such topological variants (within some scope, e.g., number of instances of a given type). These two features in combination enable automatic exploration of the space of design variants that satisfy a prescribed set of structural constraints. Nevertheless, despite these advantages, this class of method is not equipped to capture or reason about probabilistic and other quantitative aspects of system behavior (such as performance, availability).

2.2 Quantitative Verification

Quantitative Verification or Probabilistic Model Checking (PMC) [12], on the other hand, is a set of formal verification techniques that enable modeling of systems that exhibit stochastic behavior, as well as the analysis of quantitative properties that concern costs/rewards (e.g., resource usage, time) and probabilities (e.g., of violating a safety invariant).

In quantitative verification, systems are modeled as state-transition systems augmented with probabilities such as discrete-time Markov chains (DTMC), Markov decision processes (MDP), and probabilistic timed automata (PTA). System properties are expressed using some form of probabilistic temporal logic, like probabilistic computation-tree logic (PCTL), continuous stochastic logic (CSL) and probabilistic computation-tree logic extended with rewards (PRCTL)[12], [46], which state that some probability or reward meet some threshold:

- **Property Specification Languages.** An example of a probability-based PRCTL property is $P_{\geq 1}[\neg y \, U \, x]$, which captures the invariant "no request is serviced until a request has been sent" (where x and y encode that a request has been correctly issued and received, respectively). In this property, the probability quantifier P states that the path formula within the square brackets ($\neg y \, U \, x$)[8] is satisfied for a probability bound 1 (i.e., it is satisfied across all paths in the model). Probabilistic temporal logic properties are also used as specifications to reason about strategies by probabilistic model checkers [47].[9] Specifically, probabilistic model checkers enable checking for the existence of a strategy that can satisfy a threshold or optimize an objective expressed in a temporal logic formula, in systems described using formalisms that support the specification of nondeterministic choices, like MDP. For example, employing the PRCTL reward minimization operator $R^{r}_{min=?}[F \, \varphi]$ enables the synthesis of a strategy that minimizes the accrued reward r along paths that lead to states satisfying the state formula φ.
- **Modeling Languages.** Modern probabilistic model checkers like PRISM[37], COMICS [48] and STORM [38] capture models using textual languages like PRISM, PEPA, cpGCL, or low-level matrix-based descriptions. Stochastic variants of UPPAAL [49] employ graphical specification languages.

  Let us illustrate one of the most-widely used high-level languages for describing probabilistic state-transition systems, which corresponds to the probabilistic model checker PRISM, in which DTMC and MDP can be expressed as processes or modules formed by sets of commands like the following:

$$[event] \; guard \rightarrow p_1 : u_1 + \ldots + p_n : u_n$$

where *guard* is a predicate over the model variables. Each update u_i describes a transition that the process can make (by executing event) if the guard is satisfied. An update

[8] The U operator in PRCTL, read as "until" states that a given formula on the left of the operator is satisfied across all states until the first occurrence of the formula on the right of the operator.

[9] Strategies – also referred to as policies or adversaries – resolve the nondeterministic choices of a probabilistic model (e.g., MDP), selecting which action to take in every state.

is specified by giving the new values of the variables, and has a probability $p_i \in [0, 1]$. In an MDP model, multiple commands with overlapping guards introduce local nondeterminism and allow the model checker to select the alternative that resolves the nondeterministic choice in the best possible way with respect to a property expressed in probabilistic temporal logic (e.g., aimed at maximizing the probability of satisfying an invariant). In this setting, a probabilistic system model is obtained as the concurrent state machine resulting from the parallel composition of modules, which synchronize on fixed event labels. This specification style results in rigid structures of process interconnections because changing the inter-process communication topology entails manual modification of action labels across all involved processes, an error-prone and unwieldy task in non-trivial systems.

Although existing probabilistic model checkers are efficient at analyzing complex probabilistic system behaviors, their specification languages and reasoning mechanisms do not separate process instance attributes from process types, resulting in increased specification effort and hampering reusability. Furthermore, these tools are not equipped for modeling and systematically reasoning about alternative structural variants in which processes are arranged in different topologies.

2.3 Model-Based Software Engineering

Model-based Software Engineering (MBSE) [50] is an approach to system design, architecting, analysis, development, operation and maintenance of systems in which models and model transformations play the primary roles. Models are used to represent systems and to capture the aspects of interest at the right level of abstraction, while model transformations are used to manipulate models in order to extract information out of them, convert them into other models (or into code), or analyze them.

MBSE technologies also provide support for manipulating models, such as querying, transforming, merging, relating and analyzing (including the execution of) models. MBSE approaches explicitly support modularity, separation of concerns and automated generation of major system artifacts from models (for example, test cases and code), using modeling languages to represent both the systems and the operations performed on them.

MBSE will be used in the project for the definition of the high-level languages (through metamodels), the specification of the framework and its architecture, and the design and analysis of the applications. The use of MBSE will also enable the semi-automated generation of the final code of the applications using model transformations.

2.4 Search-Based Software Engineering

Search-based software engineering (SBSE) [51] is a field that applies metaheuristic search techniques such as genetic algorithms, simulated annealing and ant colony optimization to problems in software engineering, many of which can be stated as optimization problems. Optimization techniques employed in exhaustive formal verification such as linear programming or dynamic programming, which are often used in the area of probabilistic model checking (e.g., to quantify probabilities or synthesize policies for MDPs or stochastic games) are often impractical for large scale software engineering problems because of their computational complexity or their assumptions on the problem structure. Researchers and practitioners use metaheuristic search techniques, which impose little assumptions on the problem structure, to find near optimal or solutions that satisfy some minimum quality constraints.

In recent years, there have been notorious advances in employing search-based techniques to scale formal analysis of quantitative guarantees in stochastic systems by combining them with probabilistic model checking techniques [52], [53]. Although results have been promising, advance is slow because every new formalism and type of analysis requires careful tailoring of the metaheuristic search and its coordination with a formal analysis tool. In this project, we will combine the experience obtained from prior work carried out by members of the project's team in combining SBSE with formal quantitative analysis [53], as well as in MBSE to systematize and generalize the coordination of SBSE techniques with formal analysis tools.

3. Scenarios

3.1 Smart city: Urban Transport

The city of Málaga has access to a large amount of real-time information about transportation, such as the use of buses and metro lines, traffic status, etc. To improve their services, the head of the Public Transport Department (PTD) of the city has recently installed a digital twin that allows his department to use the existing information about the transport network to make predictions about peak occupancy hours using time series forecasting algorithms, identify usage patterns, and even plan the resources to optimize their use. The digital twin can also monitor the current state of the network and react to unexpected peaks or incidents by suggesting the diversion of bus routes or the addition of extra metro trains, for example.
Synthesizing such remedial plans could be done in an automated fashion from domain models in the urban digital twin that capture, among other things, the structure of the transport network and the behavior of its individual constituent components such as buses, trains, etc. Such plans, however, would need to satisfy structural constraints (e.g., a bus cannot be routed through

certain streets in without dedicated bus lanes), behavioral invariants (e.g., only one train can go through a given railway section at the same time), and quantitative constraints and optimization objectives (average passenger waiting and travel time should be minimized and not exceed a given threshold). Of course, the satisfaction of all those properties is subject to uncertainties in timing (e.g., caused by traffic conditions), resource availability (e.g., available buses and metro trains), and other aspects of the environment.

Building a scalable formal analysis and plan synthesis tool able to provide all the types of guarantees mentioned above on the generated plans would enable the PTD to mitigate the detrimental effects of adverse situations in the urban transportation system, allowing swift reaction to complex situations, e.g., by generating a repertoire of contingency plans that are readily available to be enacted whenever they are needed.

3.2 Mobile autonomous robots

In a hospital setting, medicines must be delivered to rooms and nurse stations. To give support to health workers and reduce their workload, surgical instrumentation, medicines, and other items can be transported and delivered by autonomous mobile robots equipped with secure locked drawers, which can only be open by authorized staff.[10] Some medicines are very time sensitive so the time of delivery of each medicine should be strictly controlled.

To achieve their goal, these autonomous mobile robots must carry out actions such as navigating from one location to another within the building, interacting with humans, and gaining access to spaces behind doors that might be closed. In this environment, obstacles might dynamically appear, corridors may be empty or crowded, light conditions may change, and batteries may require recharging. These robots are also limited in what they can sense, creating uncertainty in their location and chances of colliding against obstacles and walls, their speed of motion, and the resources that they may have left to complete a plan. Despite this uncertainty, they must attempt to ensure safe operation, effective use of resources (such as battery), and timeliness of accomplishing a task (e.g., for timely medicine delivery).

The critical nature of the mission of these robots demands scalable techniques and tools able to produce verified plans that respect a set of constraints that can be quantitative (e.g., the time delivery window specified for time-critical medicines should always be respected, the probability of collision against obstacles should be minimized, there should always be enough battery to achieve the robot's delivery goal and/or return to a charging station), structural (e.g., the robot should not traverse restricted areas in the hospital, only staff with the right authorization should be allowed to open the drawers of a given locker), and behavioral (e.g., no more than two robots should be moving within the same corridor at the same time).

[10] See e.g., https://vimeo.com/116978163

4. Objectives

The overarching goal of this project is defining and implementing a set of scalable methods and tools for the specification and formal analysis of domain models that include stochastic behavior descriptions, as well as structural and quantitative constraints. To deliver our vision, we will pursue the following five main objectives. These objectives will devise, validate, disseminate, and exploit the theory and tools underlying our approach and toolchain:

Objective 1 (Modelling – addresses C3) Develop methods to build user-friendly notations with mathematical underpinnings, that enable the formal modelling and reasoning about the structure, (stochastic) behavior, and quantitative aspects of systems and their components in a unified way. These notations will support the modelling of the structural, stochastic, timing, physical, and resource usage aspects of systems, as well as their behaviors, including those that entail structural changes to the system.

Objective 2 (Transformation – addresses C1 and C2) Develop methods to map models produced with the notations devised in Objective 1 to models that can be analyzed with formal analysis tools like (probabilistic) model checkers, theorem provers and constraint solvers to enable the formal verification of properties that combine structural, behavioral, and quantitative aspects of systems. These transformation methods will be formally verified to guarantee the correctness of the resulting formal models, as well as the consistency across them.

Objective 3 (Verification – addresses C4) Develop scalable methods to analyze formal guarantees on the multiple model types produced by Objective 2. This will involve devising mechanisms to orchestrate multiple formal analysis and search-based tools to achieve combined structural and behavioral/quantitative analysis at scale.

Objective 4 (Validation) To validate the theory and methods from Objectives 1-3 by using them to develop a prototype framework. To achieve this objective, we will develop two proof-of-concept instantiations in the areas of smart cities and mobile service robotics.

Objective 5 (Dissemination and exploitation) To achieve a lasting positive impact by ensuring its industrial relevance, by broadly disseminating our results, and by spearheading their adoption.

5. Work Program and Schedule

5.1 Work Program

We will pursue each of the five project objectives within a dedicated work package (WP), over a period of 36 months. For each of the work packages, we define in the following its specific objectives, tasks, deliverables, and responsible team member(s). Deliverable dates are approximate. The work group is composed by 6 people:

- The applicant, who will be involved as Principal Investigator (*PI*).
- Two academics at the department, who will be designated by the roles *P1* and *P2*.
- Three people hired by the project, with one of them developing her Ph.D. thesis in the context of the project. These individuals are assigned with roles *T1*, *T2*, and *T3* (*T1* would be the Ph.D. student).

WP0. Project management and coordination

Participants: PI (lead), P1, P2.

Objectives: Coordinate activities developed in the various WP and members of the research team, both from a technical and scientific perspective, as well as administratively.

Description: This WP will incorporate administrative and economic management tasks characteristic of a research project (acquisition of equipment and other required supplies, selection and hiring of staff, etc.), as well as scientific management and coordination (organizing project meetings, task allocation and tracking in each of the technical WP).

Tasks:

- **T0.1 Project management**. Day-to-day administration, as well as administrative and financial management of the project.
- **T0.2 Scientific management.** Scientific management will be aimed at guaranteeing that the scientific and technical objectives of the project are met according to the terms and within the agreed timeframe. The activities that will be carried out in this task include the organization of meetings and the assignment of responsibilities. Scientific management also includes activities aimed at guaranteeing agreed levels of quality with adequate risk management.
- **T0.3 Reporting.** This task will coordinate the writing of annual reports required to track progress and provide evidence of the fulfilment of established objectives and project execution.

Deliverables:

- **D0.1 Annual report, year 1**. It will report on the situation after the first year of the project, including a financial report [Month 12].
- **D0.2 Annual report, year 2**. It will report on the situation after the second year of the

project, including a financial report [Month 24].

- **D0.3 Annual report, year 3**. It will report on the situation after the third year of the project, including a financial report [Month 36].

WP1. Development of modelling methods and framework for domain models

Participants: PI (lead), P1, P2, T1, T2.

Objectives: To develop modelling notations and methods that integrate structural, behavioral, quantitative aspects of systems, and uncertainty.

Description: This WP is aligned with project Objective 1 and will be devoted to developing methods that enable the construction of languages to: (i) describe domain models that capture structure and behavior of the system, including timing, stochastic, and resource usage aspects, and (ii) capture properties that can combine structural, behavioral, and (stochastic) quantitative aspects of systems, to be analyzed on the models described with the abovementioned notations.

Tasks:

- **Task 1.1 Core concepts definition and requirements analysis:** This task will be aimed at defining the features that modelling and property languages must exhibit in order to fulfill the requirements obtained from the typical use cases derived from the scenarios considered in WP5**.**
- **Task 1.2 Language framework construction:** Informed by the results of Task 1.1, this task will be devoted to the design and development of the appropriate languages to describe domain models and their properties. We will study the suitability of various MDE tools for constructing the languages. Our shortlist includes Xtext [54]and JetBrains MPS[55].

Deliverables:

- **D1.1** Language framework specification (M06)
- **D1.2** Language framework prototype implementation (M12)

WP2. Generation of formal specifications and consistency verification methods

Participants: P1 (lead), PI, T1, T2.

Objectives: To build the mechanisms required to generate the set of formal specifications required as input to the various analysis tools, as well as to ensure their consistency.

Description: This task will be aligned with project Objective 2 and will be devoted to the development of the transformation mechanisms required to produce the inputs of each of the formal analysis tools involved in the approach, such as constraint solvers and probabilistic model checkers. This will require building metamodels for each of the target

formal specification languages, as well as model transformations to produce such formal specifications from the input domain models and properties captured in the languages devised in WP1.

Tasks:

- **Task 2.1 Metamodel definition:** This task will be aimed at defining the various metamodels required to produce the formal specifications via transformations associated with the input specifications to the disparate formal analysis tools. This will involve defining multiple metamodels for the various formalisms supported by probabilistic model checkers (e.g., DTMC, MDP, PTA), as well as for the various solvers that we plan to use, such as Alloy and Z3.
- **Task 2.2 Construction of model transformations:** This task will be devoted to building the set of model transformations required to produce the various formal specifications for probabilistic model checkers and constraint solvers. In order to do this, we will employ the Atlas Transformation Language (ATL) [56], which is currently considered a de facto standard in model-driven engineering for implementing model transformations.
- **Task 2.3 Construction of model transformation verification mechanisms:** To preserve the formal guarantees promised by the approach, the verification of model transformations is of prime importance because the correctness of the formal specifications generated (and ultimately, of the analysis mechanisms built) relies on the correctness of the operations executed using model transformations. This task will be devoted to implementing the mechanisms in charge of verifying that model transformations are correct with respect to a set of contracts whose satisfaction guarantee the consistency across formal specifications. To achieve that, we will apply and extend existing techniques for the formal analysis of model transformations developed by members of the project team in the department [57].

Deliverables:

- **D2.1 Set of formal specification metamodels** (M09)
- **D2.2 Model transformation mechanism implementation** (M15)
- **D2.3 Model transformation verification mechanism implementation** (M18)

WP3: Composition and Scalability of Formal Analyses

Participants: P2 (lead), PI, T2, T3

Objectives: To build the mechanisms required to compose different formal analysis techniques and make them scalable.

Description: This WP is aligned with project Objective 3 and is aimed at building a framework to orchestrate various formal analysis and search-based techniques with the objective of achieving combined structural and behavioral analysis at scale. Orchestrating formal

analysis and search-based techniques will involve transforming the output of some tools into an adequate format required as input to other tool, something that will be tackled using model transformation techniques like the ones described for WP2.

Tasks:

- **Task 3.1 Requirement analysis:** The purpose of this task is to gather and specify the requirements of the orchestration engine that will be used to compose formal analysis and search-based tools.
- **Task 3.2 Orchestration engine design and development:** This task will be devoted to the design and development of the orchestration engine.

Deliverables:

- **D3.1 Requirement specification of the orchestration engine** (M12)
- **D3.2 Prototype version of the orchestration engine** (M18)
- **D3.3 First stable version of the orchestration engine** (M24)
- **D3.4 Refined and tested version of the orchestration engine** (M30, M36)

WP 4: Applications and Validation

Participants: PI, P1, P2 (leads), T1, T2, T3.

Objectives: To validate the approach through evaluation of different framework instantiations.

Description: In this WP we will validate the whole project approach, including methodology and tools by developing the two instantiations of our framework that correspond to project Objective 4. In this way, we will demonstrate the feasibility and soundness of our proposal, as well as the results of the different technical tasks carried out in the project. This WP will serve also as a proof of concept showing how the proposal can be successfully applied to the development of formal analysis tools for different domains.

Tasks:

- **Task 4.1 Requirement analysis:** The purpose of this task is to gather and specify the requirements of the two scenarios described in Section 3. This task will be performed in close collaboration with Tasks 1.1 and 1.2, providing and getting feedback from them.
- **Task 4.2 Tool design and development:** This task will take advantage of the ongoing results of WPs 1, 2, and 3, tailoring them for developing the two instantiations of our framework for formal analysis.
- **Task 4.3 Evaluation:** This task will evaluate the implementation of the scenarios, which will serve both as a validation of the ongoing results of the different technical tasks and WPs and as a proof of concept of the whole approach.

Deliverables:

- **D4.1 Requirement specification of the scenarios** (M09)

- **D4.2 Prototype version of each formal analysis tool** (M18)
- **D4.3 First stable versions of each formal analysis tool** (M24)
- **D4.4 Refined and tested version of the formal analysis tools** (M30, M36)
- **D4.5 Evaluation of results and new opportunities of research and development** (M36)

WP 5: Dissemination and Exploitation

Participants: PI (lead), P1, P2, T1

Objectives: To disseminate the results of our project both in academic and industrial settings.

Description: Although this is a research-oriented project, it is expected that its results can be of interest to industrial partners who may have an interest in the project and will help with the non-academic dissemination inside their own companies and in possible projects in collaboration with our group. Technology transfer will be an important objective of the project. The project will follow Open Access policies for all produced artifacts, results and publications.

Tasks:

- **Task 5.1 Academic Dissemination:** The basic form of academic dissemination will be through high-quality journal and conference papers. Members of the project group have a previous record of publishing in top-notch journals. The main achievements of the project and its progress will be documented in a dedicated project webpage.
- **Task 5.2 Exploitation and relationship with industry:** This task will be devoted to making the results of the project visible to the industry and carried out in cooperation with potential industrial partners. We also plan to patent some of the results of the project, given its expected novelty and applicability.

Deliverables:

- **D5.1 Project Webpage** (M06)
- **D5.2 Final Exploitation Report** (M36)

5.2 Schedule

The proposed plan spans for 3 years. The chart below displays the expected schedule and duration for each WP and task, as well as the project participants that will work on each one.

		Year 1				Year 2				Year 3			
		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
WP0: Project Management	PI												
T0.1: Project management	PI												
T0.2: Scientific management	PI, P1, P2												
T0.3: Reporting	PI, P1, P2				D0.1				D0.2				D0.3
WP1: Modelling methods and framework	PI												
T1.1: Core concepts definition and req. analysis	PI, P1		D1.1										
T1.2: Language framework construction	PI, P1, T1, T2				D1.2								
WP2: Generation of formal specs and consistency ver.	P1												
T2.1: Metamodel definition	P1, PI, T1, T2			D2.1									
T2.2: Model transformation mechanisms	P1, T1, T2					D2.2							
T2.3: Model transf. verification mechanisms	P1, T1, T2						D2.3						
WP3: Composition and scalability of formal analyses	P2												
T3.1: Requirement analysis	P2, PI				D3.1								
T3.2: Orchestration engine design and development	T2, T3						D3.2		D3.3		D3.4		D3.4
WP4: Applications and Validation	PI, P1, P2												
T4.1: Requirement analysis	PI, P1, P2			D4.1									
T4.2: Tool design and development	all						D4.2		D4.3		D4.4		D4.4
T4.3: Evaluation	all						D4.2		D4.3		D4.4		D4.5
WP5: Dissemination and Exploitation	P1												
T5.1: Academic dissemination	all		D5.1										
T5.2: Exploitation and relation with industry	P1, P2, PI												D5.2

6. Methodology

The project will follow an iterative and incremental approach. This means that the WPs will overlap to allow for feedback and adaptation to the evolution of the project and the state of the art. We will use a fixed time span of six months for each iteration. A set of requirements will be assigned to the iteration following a risk-oriented approach. Each iteration will have the following phases:

- **Risk analysis and requirement selection.** Before starting an iteration, the project scope is analyzed in order to select work areas where the focus should be put in that iteration. The work areas will be selected to minimize the project risks and maximize the added value of the effort (considering dissemination and exploitation).
- **Iteration Planning.** A detailed work plan for the time span will be developed, including only those work areas that can be developed during the time span. If some expected progress is not finally achieved, it will be delayed until the next time period.
- **Development.** A conventional analysis-design-implementation cycle will be followed during this phase. In the first iterations, the analysis and design phases will have greater importance, since the developments will be oriented to clarify requirements and to build proof-of-concept components.
- **Validation.** At the end of each iteration, a validation phase will be carried out. Validation and verification activities will be of paramount importance during the project development. Validation activities will run in parallel to the development in such a way that possible problems can be discovered as soon as possible.

7. Risks and Contingency plans

The development of the project will follow a risk-oriented methodology whenever it is possible. This means that the most critical activities have been planned to be carried out in the initial stages of the process, in such a way that the appropriate measures can be taken as soon as possible in the project planning. Anyway, the following issues have been identified as primary risks for the project:

- The rapid evolution of underlying technologies. Formal verification techniques and tools evolve at a steady pace. New systems, standards and tools will probably emerge during the next few years, and they can affect the methodologies and tools developed in this project. The implication of some of the members of the project team in the international research community in the area will try to minimize these effects.
- Inability to define models to capture and analyze combined structural, behavioral, and quantitative aspects of system models. The project incorporates leading-edge theoretical research, whose absolute success cannot be predicted. Although we strongly believe that it is entirely feasible to find abstract languages and models for the kind of concepts and mechanisms that we want to express, the intrinsic nature of the research objectives makes it impossible to assure that such languages and models can be found in the context of the project. In any case, our exploratory research mentioned in Section 1 indicates that there are reasonable expectations of feasibility. However, should the team find itself in the worst-case scenario, the results of the project will be adequately reported to the scientific community so that the community can benefit from the project. In order to reduce risk, we will try to reuse and adapt existing languages in related domains, trying to avoid creating completely new languages from scratch.
- Lack of tooling and technologies. We may develop our own tools or extend already existing ones, which may represent a heavy overload for the project, due to the effort required to build tools. We expect to be able to reuse some of the existing tools and technologies, extending them accordingly, but the lack of maturity of this field represents a challenge that cannot be neglected. The experience of our group in building tools is an asset that we expect can mitigate this risk.

8. Coordination Meetings and Problem Resolution

Given the physical proximity of the team members, there will be continuous contact among them. In any case, and for coordination purposes, periodic follow-up meetings have been scheduled every three months. In these meetings, each WP leader will report on the status and

level of achievements of his objectives, and corrective actions will be agreed upon in case of potential delays.
Generally, all issues and problems found will be dealt with at the WP level first, and then reported at the periodical coordination meetings if solved. WP leaders will also be in charge of coordinating with other WP leaders in case some problems are detected that may affect other WPs. The PI will always be informed of these issues, not only for coordination and project management purposes but also for recording all detected problems.

9. Scientific and Technical Impact

Beyond foundational contributions to the important emerging area of quantitative verification, we anticipate that the results of this work will have a broader impact on software engineering by showing how lightweight formal methods can augment the set of tools available to address the verification of quantitative aspects of correctness in increasingly complex software-intensive systems that operate under uncertain conditions. Such combination of techniques has been largely unexplored, although we believe that it has the potential for significant impact. The ability to systematically verify alternative system designs under uncertainty and changes in their structure that happen at run time could have a significant and far-reaching impact on the quality of the systems on which we increasingly rely (autonomous, self-adaptive, socio-cyber-physical).
This research has the potential to lay the foundations to provide software designers with tools to make design decisions in poorly understood design spaces. These spaces involve uncertainties concerning, for example, the reliability of components, lack of predictability of the operation environment, or even interactions with humans. Designers typically rely on intuition to navigate these design spaces but getting these wrong can yield systems that lead to serious failures and fall short of meeting market needs with the qualities that are desired by users. Furthermore, designers may miss novel designs that would be more elegant and resilient to changes over time.
Additional broader impacts will occur in the area of education. The results of this research will inform the content of several courses in our graduate curriculum that treat formal methods, software architecture, and adaptive systems. Further, PI and P1 will jointly advise a Ph.D. student. We plan to make the tools developed in this research available to the broader educational and research community (via a dedicated public website, public GitHub repositories, and other distribution channels like self-adaptive.org), so that others can build on our work and leverage it in their teaching.

10. Budget

The Budget for our three-year project proposal is divided in two parts, which concern staff and execution costs, respectively. In the part that corresponds to staff, we request two people hired by the university for 36 months. The first one (T1) is intended to carry out her Ph.D. thesis in the context of the project and would work on development tasks corresponding to WPs 1, 2, and 4, whereas the second one (T2) would work on WPs 1, 2, 3, and 4. There would be an additional technical staff member (T3) hired to by the university to work on WP3 for 27 months (starting in month 9).

Execution costs are described in the table of Section 10.1, which we present in the following.

10.1 Staff Cost

We assume that a Ph.D. student receives a stipend of 20.000 EUR per annum, and a technician with university-level qualifications receives a stipend of 27.000 EUR per annum (gross).

ID	Profile	Tasks	Cost (EUR)
T1	Graduate student (Ph.D.)	36 months for development tasks in WP1, WP2, WP4	60.000
T2	Support technical staff	36 months for development tasks in WP1, WP2, WP3, WP4	81.000
T3	Support technical staff	27 months for development tasks in WP3, WP4	60.750
Total staff costs			**201.750**

10.2 Execution Cost

Description	Justification	Cost (EUR)
Materials(non-consumable)		
Desktop computer	A desktop computer with adequate graphics card and display for visualization	2.000
Laptop computers	Three computers for hired researchers	3.000
Consumables		
Misc. consumables		1.500
Other		
Software licenses	Modeling software and other licenses	3.000
Travel and subsistence		
Conference attendance (national and international)	We estimate 6 attendances to international conference, with an average cost of 1000 EUR, and 6 attendances to national events, with an average cost of 500 EUR.	9.000
Attendance to events to favor internationalization and dissemination	To collaborate with other groups and participating in preparing international initiatives	3.500

Expert visit	To the purpose of evaluating progress, we will invite an expert (one visit per year) in the area of the proposal	2.500
Conference and event registration fees		
Registration fees	Partial funding for conference and (inter)national event registration fees	5.000
Dissemination costs		
Workshop organization	Organization of a local, a national, and an international workshop	4.500
Other		
Auditing	Auditing	1.200
Total execution cost		**35.200**

10.3 Indirect Cost

Indirect costs	**30.262**

Total project cost: 201.750 + 35.200 + 30.263 = 267.213

Bibliografía

[1] T. A. Henzinger, "From Boolean to Quantitative Notions of Correctness," in *Proceedings of the 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2010, pp. 157–158. doi: 10.1145/1706299.1706319.

[2] R. Calinescu, C. Ghezzi, M. Kwiatkowska, and R. Mirandola, "Self-Adaptive Software Needs Quantitative Verification at Runtime," *Commun. ACM*, vol. 55, no. 9, pp. 69–77, Sep. 2012, doi: 10.1145/2330667.2330686.

[3] J. Cámara, D. Garlan, and B. R. Schmerl, "Synthesizing tradeoff spaces with quantitative guarantees for families of software systems," *J. Syst. Softw.*, vol. 152, pp. 33–49, 2019, doi: 10.1016/j.jss.2019.02.055.

[4] J. Cámara, B. R. Schmerl, G. A. Moreno, and D. Garlan, "MOSAICO: offline synthesis of adaptation strategy repertoires with flexible trade-offs," *Autom. Softw. Eng.*, vol. 25, no. 3, pp. 595–626, 2018, doi: 10.1007/s10515-018-0234-9.

[5] G. A. Moreno, J. Cámara, D. Garlan, and B. R. Schmerl, "Flexible and Efficient Decision-Making for Proactive Latency-Aware Self-Adaptation," *ACM Trans. Auton. Adapt. Syst.*, vol. 13, no. 1, pp. 3:1–3:36, 2018, doi: 10.1145/3149180.

[6] J. Cámara, R. de Lemos, N. Laranjeiro, R. Ventura, and M. Vieira, "Robustness-Driven Resilience Evaluation of Self-Adaptive Software Systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 14, no. 1, pp. 50–64, 2017, doi: 10.1109/TDSC.2015.2429128.

[7] J. Cámara *et al.*, "Incorporating architecture-based self-adaptation into an adaptive industrial software system," *J. Syst. Softw.*, vol. 122, pp. 507–523, 2016, doi: 10.1016/j.jss.2015.09.021.

[8] J. Cámara, A. Lopes, D. Garlan, and B. R. Schmerl, "Adaptation impact and environment models for architecture-based self-adaptive systems," *Sci. Comput. Program.*, vol. 127, pp. 50–75, 2016, doi: 10.1016/j.scico.2015.12.006.

[9] J. Cámara, G. A. Moreno, D. Garlan, and B. R. Schmerl, "Analyzing Latency-Aware Self-Adaptation Using Stochastic Games and Simulations," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 4, pp. 23:1–23:28, 2016, doi: 10.1145/2774222.

[10] J. Cámara, R. de Lemos, M. Vieira, R. Almeida, and R. Ventura, "Architecture-based resilience evaluation for self-adaptive systems," *Computing*, vol. 95, no. 8, pp. 689–722, 2013, doi: 10.1007/s00607-013-0311-7.

[11] J. Cámara, W. Peng, D. Garlan, and B. R. Schmerl, "Reasoning about sensing uncertainty and its reduction in decision-making for self-adaptation," *Sci. Comput. Program.*, vol. 167, pp. 51–69, 2018, doi: 10.1016/j.scico.2018.07.002.

[12] M. Z. Kwiatkowska, G. Norman, and D. Parker, "Stochastic Model Checking," in *Formal Methods for Performance Evaluation, 7th International School on Formal*

Methods for the Design of Computer, Communication, and Software Systems, SFM, 2007, vol. 4486, pp. 220–270.

[13] J. Cámara, P. Correia, R. de Lemos, and M. Vieira, "Empirical resilience evaluation of an architecture-based self-adaptive software system," in *QoSA'14, Proceedings of the 10th International ACM SIGSOFT Conference on Quality of Software Architectures (part of CompArch 2014), Marcq-en-Baroeul, Lille, France, June 30 - July 04, 2014*, 2014, pp. 63–72. doi: 10.1145/2602576.2602577.

[14] D. Jackson, "Alloy: a lightweight object modelling notation," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 2, pp. 256–290, 2002.

[15] J. Cámara, "HaiQ: Synthesis of Software Design Spaces with Structural and Probabilistic Guarantees," in *FormaliSE@ICSE 2020: 8th International Conference on Formal Methods in Software Engineering, Seoul, Republic of Korea, July 13, 2020*, 2020, pp. 22–33. doi: 10.1145/3372020.3391562.

[16] J. Cámara, D. Garlan, and B. R. Schmerl, "Synthesis and Quantitative Verification of Tradeoff Spaces for Families of Software Systems," in *Software Architecture - 11th European Conference, ECSA 2017, Canterbury, UK, September 11-15, 2017, Proceedings*, 2017, vol. 10475, pp. 3–21.

[17] J. Warmer and A. Kleppe, *The Object Constraint Language: Getting Your Models Ready for MDA*. Addison-Wesley, 2003.

[18] J. M. Spivey, *Z Notation - a reference manual (2. ed.)*. Prentice Hall, 1992.

[19] J.-R. Abrial, M. K. O. Lee, D. Neilson, P. N. Scharbach, and I. H. Sørensen, "The B-Method," in *VDM '91 - Formal Software Development*, 1991, vol. 552, pp. 398–405.

[20] D. Bjørner, "The Vienna development method (VDM): Software specification & program synthesis," in *Mathematical Studies of Information Processing, Proceedings of the International Conference*, 1978, vol. 75, pp. 326–359.

[21] S. Maoz, J. O. Ringert, and B. Rumpe, "Synthesis of component and connector models from crosscutting structural views," in *European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE'13*, 2013, pp. 444–454.

[22] S. Wong, J. Sun, I. Warren, and J. Sun, "A Scalable Approach to Multi-style Architectural Modeling and Verification," in *13th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2008)*, 2008, pp. 25–34.

[23] H. Bagheri, C. Tang, and K. J. Sullivan, "TradeMaker: automated dynamic analysis of synthesized tradespaces," in *36th Int. Conf. on Software Engineering*, 2014, pp. 106–116.

[24] P. Zave, "A Formal Model of Addressing for Interoperating Networks," in *FM 2005: Formal Methods, International Symposium of Formal Methods Europe*, 2005, vol. 3582, pp. 318–333.

[25] H. Bagheri, A. Sadeghi, J. Garcia, and S. Malek, “COVERT: Compositional Analysis of Android Inter-App Permission Leakage,” *IEEE Trans. Software Eng.*, vol. 41, no. 9, pp. 866–886, 2015.

[26] D. Garlan, “Software engineering in an uncertain world,” in *Proceedings of the Workshop on Future of Software Engineering Research, FoSER 2010, at the 18th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2010, pp. 125–128.

[27] A. Filieri, C. Ghezzi, and G. Tamburrelli, “Run-time efficient probabilistic model checking,” in *Proceedings of the 33rd International Conference on Software Engineering, ICSE*, 2011, pp. 341–350.

[28] F. Arbab, S. Meng, Y.-J. Moon, M. Z. Kwiatkowska, and H. Qu, “Reo2MC: a tool chain for performance analysis of coordination models,” in *Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2009, Amsterdam, The Netherlands, August 24-28, 2009*, 2009, pp. 287–288.

[29] L. Gilmore Stephen and Kloul, “A Unified Tool for Performance Modelling and Prediction,” in *Computer Safety, Reliability, and Security: 22nd International Conference, SAFECOMP 2003, Edinburgh, UK, September 23-26, 2003. Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 179–192.

[30] R. Calinescu, L. Grunske, M. Z. Kwiatkowska, R. Mirandola, and G. Tamburrelli, “Dynamic QoS Management and Optimization in Service-Based Systems,” *IEEE Trans. Software Eng.*, vol. 37, no. 3, pp. 387–409, 2011.

[31] F. and L. S. Kuntz Matthias and Leitner-Fischer, “From Probabilistic Counterexamples via Causality to Fault Trees,” in *Computer Safety, Reliability, and Security: 30th International Conference,SAFECOMP 2011, Naples, Italy, September 19-22, 2011. Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 71–84.

[32] S. Basagiannis, S. Petridou, N. Alexiou, G. Papadimitriou, and P. Katsaros, “Quantitative analysis of a certified e-mail protocol in mobile environments: A probabilistic model checking approach,” *Computers and Security*, vol. 30, no. 4, pp. 257–272, 2011.

[33] R. Calinescu, M. Ceska, S. Gerasimou, M. Kwiatkowska, and N. Paoletti, “Designing Robust Software Systems through Parametric Markov Chain Synthesis,” in *2017 IEEE International Conference on Software Architecture, ICSA 2017, Gothenburg, Sweden, April 3-7, 2017*, 2017, pp. 131–140.

[34] A. Naskos, A. Gounaris, H. Mouratidis, and P. Katsaros, “Online Analysis of Security Risks in Elastic Cloud Applications,” *IEEE Cloud Computing*, vol. 3, no. 5, pp. 26–33, 2016.

[35] M. Akbarzadeh and M. A. Azgomi, “A framework for probabilistic model checking of security protocols using coloured stochastic activity networks and PDETool,” in *2010 5th International Symposium on Telecommunications*, 2010, pp. 210–215.

[36] S. Basagiannis, P. Katsaros, A. Pombortsis, and N. Alexiou, “Probabilistic model checking for the quantification of DoS security threats,” *Computers and Security*, vol. 28, no. 6, pp. 450–465, 2009.

[37] M. Z. Kwiatkowska, G. Norman, and D. Parker, “PRISM 4.0: Verification of Probabilistic Real-Time Systems,” in *Computer Aided Verification - 23rd International Conference, CAV*, 2011, vol. 6806, pp. 585–591.

[38] C. Dehnert, S. Junges, J.-P. Katoen, and M. Volk, “A storm is Coming: A Modern Probabilistic Model Checker,” *CoRR*, vol. abs/1702.04311, 2017, [Online]. Available: http://arxiv.org/abs/1702.04311

[39] S. Gilmore and J. Hillston, “The PEPA Workbench: A Tool to Support a Process Algebra-based Approach to Performance Modelling,” in *Computer Performance Evaluation, Modeling Techniques and Tools, 7th International Conference*, 1994, vol. 794, pp. 353–368.

[40] H. Jifeng, K. Seidel, and A. McIver, “Probabilistic models for the guarded command language,” *Science of Computer Programming*, vol. 28, no. 2, pp. 171–192, 1997.

[41] N. Esfahani, S. Malek, and K. Razavi, “GuideArch: guiding the exploration of architectural solution space under uncertainty,” in *35th International Conference on Software Engineering, ICSE*, 2013, pp. 43–52.

[42] F. Brosch, H. Koziolek, B. Buhnova, and R. H. Reussner, “Architecture-Based Reliability Prediction with the Palladio Component Model,” *IEEE Trans. Software Eng.*, vol. 38, no. 6, pp. 1319–1339, 2012.

[43] D. Perez-Palacin and R. Mirandola, “Uncertainties in the Modeling of Self-adaptive Systems: A Taxonomy and an Example of Availability Evaluation,” in *Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering*, 2014, pp. 3–14.

[44] L. de Moura and N. Bjørner, “Z3: An efficient SMT Solver,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, vol. 4963 LNCS. doi: 10.1007/978-3-540-78800-3_24.

[45] J.-R. Abrial, *Modeling in Event-B - System and Software Engineering*. Cambridge University Press, 2010.

[46] S. Andova, H. Hermanns, and J.-P. Katoen, “Discrete-Time Rewards Model-Checked,” in *Formal Modeling and Analysis of Timed Systems: First International Workshop, FORMATS*, 2003, vol. 2791, pp. 88–104.

[47] M. Z. Kwiatkowska and D. Parker, “Automated Verification and Strategy Synthesis for Probabilistic Systems,” in *Automated Technology for Verification and Analysis - 11th International Symposium, ATVA 2013, Hanoi, Vietnam, October 15-18, 2013. Proceedings*, 2013, vol. 8172, pp. 5–22.

[48] E. and V. M. and W. R. and K. J.-P. and B. B. Jansen Nils and Ábrahám, “The COMICS Tool – Computing Minimal Counterexamples for DTMCs,” in *Automated Technology for Verification and Analysis: 10th International Symposium, ATVA 2012, Thiruvananthapuram, India, October 3-6, 2012. Proceedings*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 349–353.

[49] A. David, P. G. Jensen, K. G. Larsen, M. Mikučionis, and J. H. Taankvist, “Uppaal Stratego,” in *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 9035, Springer Berlin Heidelberg, 2015, pp. 206–211.

[50] M. Brambilla, J. Cabot, and M. Wimmer, “Model-Driven Software Engineering in Practice: Second Edition,” *Synthesis Lectures on Software Engineering*, vol. 3, no. 1, 2017, doi: 10.2200/s00751ed2v01y201701swe004.

[51] M. Harman and B. F. Jones, “Search-based software engineering,” *Information and Software Technology*, vol. 43, no. 14, 2001, doi: 10.1016/S0950-5849(01)00189-6.

[52] S. Gerasimou, R. Calinescu, and G. Tamburrelli, “Synthesis of probabilistic models for quality-of-service software engineering,” *Automated Software Engineering*, vol. 25, no. 4, 2018, doi: 10.1007/s10515-018-0235-8.

[53] S. Gerasimou, J. Cámara, R. Calinescu, N. Alasmari, F. Alhwikem, and X. Fang, “Evolutionary-Guided Synthesis of Verified Pareto-Optimal MDP Policies,” 2021.

[54] M. Eysholdt and H. Behrens, “Xtext - Implement your language faster than the quick and dirty way tutorial summary,” 2010. doi: 10.1145/1869542.1869625.

[55] M. Voelter and V. Pech, “Language modularity with the MPS language workbench,” 2012. doi: 10.1109/ICSE.2012.6227070.

[56] F. Jouault, F. Allilaire, J. Bézivin, and I. Kurtev, “ATL: A model transformation tool,” *Science of Computer Programming*, vol. 72, no. 1–2, 2008, doi: 10.1016/j.scico.2007.08.002.

[57] B. J. Oakes, J. Troya, L. Lúcio, and M. Wimmer, “Full contract verification for ATL using symbolic execution,” *Software and Systems Modeling*, vol. 17, no. 3, 2018, doi: 10.1007/s10270-016-0548-7.

PROYECTO DOCENTE

JAVIER CÁMARA MORENO

PRUEBA DE ACCESO AL CUERPO DE PROFESOR TITULAR DE UNIVERSIDAD
DEL ÁREA DE LENGUAJES Y SISTEMAS INFORMÁTICOS

Málaga, 27 de enero de 2022

Preámbulo

Por resolución de 9 de noviembre de 2021, BOE núm. 277 de 19 de noviembre de 2021, la Universidad de Málaga convocó a concurso de acceso, entre otras, la plaza de Profesor Titular de Universidad cuyos detalles se muestran a continuación. El presente documento se ha elaborado para cumplir parcialmente con las instrucciones establecidas en dicha convocatoria en relación con la documentación a entregar en el acto de presentación del concurso, a saber:

"*Proyecto docente, por sextuplicado, referido a una asignatura obligatoria adscrita al área de conocimiento de la plaza objeto de concurso e incluida en el correspondiente plan de estudios de una titulación oficial de Grado o Máster de la Universidad de Málaga, con validez en todo el territorio nacional y que esté vigente en la fecha de publicación de la convocatoria de la plaza en el BOE."*

Cuerpo	Profesor Titular de Universidad
Número de plaza	037TUN21
Área de conocimiento	Lenguajes y Sistemas Informáticos
Departamento	Lenguajes y Ciencias de la Computación
Perfil docente	Docencia en Introducción a la Ingeniería del Software, asignatura adscrita al Área de Conocimiento
Perfil investigador	Investigación en s*oftware system structures* y *software functional properties*.

ÍNDICE

PARTE I. CONTEXTO

Uno de los criterios considerados en la provisión de plazas de profesorado universitario es la adecuación del proyecto docente a las necesidades de la universidad en la que se pretende desarrollar. Por este motivo, se analizan en este capítulo la situación actual de la Universidad de Málaga, de la Escuela Técnica Superior de Ingeniería Informática ― que es el centro en el que se oferta la asignatura objeto de este proyecto― y del Departamento de Lenguajes y Ciencias de la Computación, responsable de la docencia de esta asignatura.

Así, en este capítulo se analiza el contexto general en el que se va a llevar a cabo el presente proyecto docente, incluyendo los centros y organismos responsables de las asignaturas, los planes de estudio que las definen y las trayectorias curriculares donde se enmarcan.

1. Contexto institucional

La Universidad tiene la función de la educación superior mediante la docencia y la investigación. Una de las razones fundamentales de su existencia es, por tanto, dar una formación integral y de calidad a los estudiantes. La Universidad debe ser un motor importante de la mejora de la sociedad y del crecimiento económico. Dada la naturaleza investigadora de la Universidad, es especialmente importante su dedicación a la investigación y al desarrollo tecnológico. Respecto a la función investigadora, la Universidad ― y en especial una Escuela Técnica Superior― debe ser un núcleo de actualización de técnicas, y debe ofrecer un continuo servicio de reciclaje, tanto para los profesores como para los profesionales. Por ello, la investigación en la Universidad deberá cubrir no solamente la tarea encomendada a la investigación en general ― tal como puede ser entendida en los centros de investigación pura― , sino que además deberá completar la labor docente y educativa del profesor. De la misma forma, es su competencia estar en vanguardia de las últimas innovaciones científico-técnicas.

A esta tradición se une en la actualidad una frecuente atención a los procesos de mejora de la gestión y de la docencia, gracias a la utilización de las tecnologías de la información y la comunicación. De esta evolución deben hacerse cargo las Escuelas de Ingeniería en su afán de poner al servicio de la sociedad los grandes recursos tecnológicos que están en continua evolución.

La necesaria interconexión entre ingenieros y ciencia debe formarse y desarrollarse en la etapa universitaria de aquellos. Esta consideración de la Universidad como ámbito adecuado para la formación científica del ingeniero viene de lejos. El desarrollo económico de un país parece depender fuertemente de la capacidad de sus ingenieros para afrontar estos planteamientos.

Hay dos retos que las universidades están afrontando en los últimos años y a los que deberán seguir atentos: la innovación y la calidad. Desde hace más de una década, las universidades han ido poniendo en marcha distintos sistemas que aseguren:

- la calidad de los profesores, a través de diversos sistemas de selección, promoción y nombramiento,
- la calidad de los estudiantes, a través de sistemas de admisión selectiva, exámenes y pruebas, becas y premios,
- la calidad de la investigación, a través de financiación y ayudas a los proyectos de investigación, índices de citas y publicaciones, evaluaciones internas y externas periódicas,
- la calidad de los planes de estudio, de los cursos y de la enseñanza en general, a través de la evaluación o acreditación de programas y de cursos, de cuestionarios a los estudiantes, de evaluación del profesorado y de programas de formación en habilidades docentes,
- la calidad del personal de administración y servicios, a través de los diversos sistemas de selección y formación.

De la misma forma, todas las universidades han apostado por la innovación en las metodologías de enseñanza-aprendizaje y se han puesto en marcha distintas iniciativas que han permitido, entre otras cosas, la incorporación de las nuevas tecnologías en la práctica docente, contribuyendo a una transformación de las formas de interacción en la comunidad universitaria. Así, en lo que respecta a la adaptación de la actividad universitaria a este contexto emergente, un objetivo innovador no pasa tanto por la integración del ordenador en el aula como por la transformación y permeabilidad de la universidad ante las prácticas digitales que están surgiendo a partir del uso y apropiación de las nuevas tecnologías; especialmente en lo que concierne a los procesos de construcción, intercambio y divulgación del conocimiento que definen lo que es o ha de ser la propia universidad del siglo XXI y de la era digital.

Esta transformación requiere un cuidadoso proceso de reflexión y análisis crítico. Los cambios en las metodologías y enfoques docentes, así como la introducción de nuevas herramientas y estrategias pedagógicas en los procesos de enseñanza-aprendizaje deberán estar orientados a la educación de personas capaces de manejar tecnologías, conceptos y lenguajes cambiantes, que les sirvan para adaptarse a los cambios permanentes de la sociedad contemporánea. Las universidades están dando respuesta, en el ámbito de la innovación, de distintas formas:

- Poniendo en marcha procesos de adaptación al nuevo contexto de una sociedad en la que las tecnologías digitales modifican la construcción, intercambio y divulgación del saber.
- Promoviendo una dinámica de adaptación permanente a los cambios rápidos y constantes, así como a las lógicas de participación que se dan en la sociedad.
- Incrementando y fomentando el desarrollo de las competencias y habilidades digitales en el conjunto de la comunidad universitaria: estudiantes, profesorado y personal de administración y servicios.
- Sirviendo de puente en todo lo relativo a procesos de innovación docente entre el mundo universitario, la realidad del mundo de la empresa y el trabajo.

Las universidades deben poner todos los recursos para mantener estos objetivos de calidad e innovación y facilitar los mecanismos que los aseguren y mejoren. Actualmente, se está imponiendo un modelo de calidad e innovación relacionado con la capacidad de transformación y cambio de la Universidad. Para ello, se suele poner especial énfasis en la necesidad de gestión del cambio ante las nuevas demandas de la sociedad, de introducción de la tecnología actual y de adaptación de los recursos humanos.

1.1. La Universidad

La LOU (Ley Orgánica 6/2001, de 21 de diciembre, BOE núm. 307/2001, de 24 de diciembre) fue aprobada por las Cortes Generales el 21 de diciembre de 2001 y modificada el 12 de abril de 2007 (Ley Orgánica 4/2007, de 12 de abril, BOE de 13 de abril de 2007). Los aspectos principales de la LOU, según se precisa en su preámbulo, se pueden resumir en los siguientes puntos:

- Tiene como objetivo mejorar la calidad del sistema universitario.
- Se establece un sistema de selección del profesorado más abierto, competitivo y transparente que garantice el mérito y la capacidad.
- Se establecen mecanismos para impulsar el desarrollo de la investigación.
- Facilita que la gestión de las universidades sea más ágil y eficaz.
- Introduce mecanismos de evaluación de la calidad desarrollados a través de la creación de la Agencia Nacional de Evaluación y Acreditación de la Calidad (ANECA).
- Se impulsa un mayor acercamiento entre la Universidad y la sociedad.

- Se refuerzan las competencias de los Consejos Sociales para que el conjunto de la sociedad participe más activamente en la vida universitaria.

La LOU preparaba al sistema universitario español para su incorporación al espacio universitario europeo.

Órganos de dirección y representación

La LOU pretende facilitar una gestión más ágil y eficaz de las universidades, para lo cual se establece una diferenciación entre órganos de dirección o gestión y órganos de representación y supervisión. Los principales órganos, unipersonales o colectivos, establecidos son los siguientes:

Consejo de Gobierno. Es el máximo órgano de gobierno de una universidad. Le corresponde la aprobación de las líneas estratégicas y programáticas en lo que se refiere a los recursos humanos, investigación, organización de las enseñanzas, recursos económicos y elaboración de los presupuestos. Tiene como miembros natos al Rector (que lo preside), al Gerente y al Secretario General. El Consejo de Gobierno tiene un máximo de 40 miembros, elegidos un 30% por el Rector, un 40% por el Claustro y el 30% restante ― según los Estatutos― de entre decanos, directores de escuelas, institutos, etc. También forman parte el presidente y dos representantes del Consejo Social.

Rector. Su figura sale reforzada al conferirle un carácter más presidencialista. Se rediseñan sus competencias y las de su Equipo de Gobierno. El Rector es elegido por la comunidad universitaria mediante sufragio universal, libre y secreto, con una ponderación de voto que representa a los distintos estamentos.

Junta Consultiva. Para lograr un mejor desarrollo de las funciones que se encomiendan al Equipo de Gobierno, éste cuenta con una Junta Consultiva, que tiene funciones de consulta y asesoramiento. Estará constituida por profesores e investigadores de reconocido prestigio, con méritos docentes e investigadores acreditados por las correspondientes evaluaciones positivas.

Consejo Social. La LOU refuerza las competencias y funciones del Consejo Social para perfeccionar el cumplimiento de las tareas de supervisión y de rendición de cuentas de la Universidad. Estas competencias son, entre otras: supervisión de actividades económicas; aprobación del presupuesto y programación plurianual; dar conformidad a la propuesta del Rector para el nombramiento del Gerente; acordar complementos retributivos del profesorado; supervisión del desarrollo y ejecución de los presupuestos; y aprobar la creación de Fundaciones y otras personas jurídicas. Su composición y funciones se regulan por ley de la Comunidad Autónoma, si bien deben ser miembros fijos el Rector, el

Secretario General y el Gerente, además de un profesor, un estudiante y un representante del personal de administración y servicios.

Claustro. Es el máximo órgano de representación de la comunidad universitaria. Está presidido por el Rector. Le corresponden, entre otras funciones, la elaboración de los estatutos y la elección del 40% del Consejo de Gobierno. También tiene la potestad de ejercer control sobre el Rector y, en determinados casos, puede proponer la convocatoria de elecciones. El Claustro tiene una composición mayoritaria de profesores funcionarios doctores (51%), siendo el 49% restante determinado por los estatutos de cada universidad.

El profesorado

En cuanto a la composición del profesorado, la norma establece que las universidades públicas podrán contar con profesores funcionarios o contratados, si bien los primeros siempre deberán ser al menos el 51%. La Ley pretende un equilibrio adecuado entre las funciones docentes e investigadoras que definen y conforman la esencia del profesor de universidad. Para ello, se comienza por una primera categoría de ayudante, contratado por un máximo de cuatro años entre aquellos que ya hubieran cursado los créditos de doctorado, cuya finalidad principal será la de completar su formación científica. El siguiente paso es el profesor ayudante doctor, contratado también por un máximo de cuatro años. El proyecto establece que para acceder a la categoría de profesor ayudante doctor se requiere una evaluación externa positiva y será necesario no haber tenido vinculación con la universidad contratante en los dos años anteriores. Con este requisito se persigue propiciar la movilidad.

Por otra parte, se crea la figura del profesor contratado doctor, que desarrollará tareas de docencia y de investigación, o prioritariamente de investigación. Esta figura está reservada a los doctores que acrediten al menos tres años de actividad postdoctoral docente e investigadora, o prioritariamente investigadora, y hayan recibido una evaluación externa positiva. La figura de profesor asociado se restringe sólo para contratados a tiempo parcial.

Finalmente, según la reforma de abril de 2007, la selección del profesorado funcionario se modifica incorporando un modelo de acreditación que permita que las universidades seleccionen a su profesorado entre los previamente acreditados. Este sistema incorpora para el conjunto de la comunidad académica un mayor rigor en la acreditación y una mayor flexibilidad para las universidades en la selección de su personal.

Los estudiantes

En la LOU se incluye un título específico relacionado con los estudiantes. En él se tratan los sistemas de acceso a la universidad, la oferta de plazas, los derechos y deberes de los estudiantes y las ayudas y becas al estudio. En cuanto al gobierno de las universidades, la

elección directa del Rector permite una participación directa de los estudiantes en el proceso. Su participación en los restantes órganos de gobierno se deja en manos de los Estatutos, que son los que determinan el porcentaje de representación que tienen. Por otra parte, se implanta el distrito único abierto para facilitar la movilidad de los estudiantes entre universidades.

Evaluación de la calidad

Un aspecto novedoso incluido en la LOU fueron los programas de evaluación y acreditación para promover e impulsar la calidad de las universidades. Con este fin, se creó la ANECA, cuyos cometidos son:

- Medir y hacer público el rendimiento del servicio universitario, facilitando así la rendición de cuentas a la sociedad.
- Proporcionar información a los estudiantes, a las familias y al conjunto de la sociedad sobre la calidad de los centros universitarios.
- Impulsar la competitividad, comparación y transparencia de las universidades.
- Potenciar la mejora de la calidad docente, investigadora y de gestión.
- Aportar información cualificada a las administraciones, fundamental para la toma de decisiones.
- Fomentar la competitividad de las universidades españolas en el ámbito europeo e internacional

1.2. La Universidad de Málaga

La Universidad de Málaga fue creada por el Decreto 2566/72 del 18 de agosto de 1972, agrupando la Facultad de Ciencias Económicas y Empresariales (hasta entonces integrada en la Universidad de Granada) y la Facultad de Medicina. En su inicio contaba con apenas tres mil alumnos, que se añadían a los de las escuelas ya existentes de Peritos, Comercio y Magisterio.

Desde su creación, ha sufrido un crecimiento continuo, ya que se han ido incorporando nuevos estudios. En el curso 2020/21 se han ofertado 76 titulaciones de grado, 79 másteres universitarios y 22 programas de doctorado. Por otra parte, en la actualidad se compone de 15 Facultades, 5 Escuelas Técnicas Superiores, tres Centros Adscritos y una Escuela de Doctorado. La ubicación de estos centros se reparte entre los campus de El Ejido y de Teatinos, además de los centros adscritos de la Diputación, Ronda y Antequera.

Según el análisis global de la Universidad de Málaga correspondiente al curso 2018/19, el total de alumnos matriculados era 35.876, y la plantilla constaba de 2499 profesores. Las enseñanzas relativas a las Tecnologías de la Información se imparten la Escuela Técnica Superior de Ingeniería Informática.

1.3. La Escuela Técnica Superior de Ingeniería Informática

Los orígenes de los estudios en Tecnologías de la Información en la Universidad de Málaga se remontan a 1982, cuando por el Real Decreto 1619, de 18 de junio, se crea la Sección de Informática dentro de la ya existente Escuela Universitaria Politécnica. En dicha Sección de Informática se impartían los estudios de Diplomado en Informática. Unos años después, en 1987 (Decreto 208/1987) se crea la Facultad de Informática, en la que se impartieron — hasta el curso 1993/94— los estudios conducentes a la obtención del título de Licenciado en Informática.

En 1994 se crea la Escuela Técnica Superior de Ingeniería Informática, mediante la fusión de las citadas Facultad de Informática y Sección de Informática de la Escuela Universitaria Politécnica. Desde el año 1995 está ubicada en el Complejo Tecnológico de la Universidad de Málaga, ubicación que comparte con la E.T.S. de Ingeniería de Telecomunicación. Los estudios que se impartían en aquel entonces, hasta la llegada de las actuales titulaciones de Grado y Máster, eran los de Ingeniero en Informática, Ingeniero Técnico en Informática de Gestión e Ingeniero Técnico en Informática de Sistemas.

Actualmente la Escuela Técnica Superior de Ingeniería Informática imparte los estudios conducentes a los Grados en Ingeniería Informática, Ingeniería del Software — en el que se encuadra la asignatura objeto de este proyecto docente—, Ingeniería de Computadores, e Ingeniería de la Salud, titulación conjunta de la Universidad de Málaga y la Universidad de Sevilla, así como el Máster en Ingeniería Informática, el Master en Ingeniería del Software e Inteligencia Artificial y el Programa de Doctorado en Tecnologías Informáticas.

1.4. El Departamento de Lenguajes y Ciencias de la Computación

El Departamento de Lenguajes y Ciencias de la Computación se creó en 1989 y es responsable de la docencia e investigación en tres áreas de conocimiento:

- Lenguajes y Sistemas Informáticos.
- Ciencias de la Computación e Inteligencia Artificial.
- Ingeniería Telemática.

A la primera de ellas, Lenguajes y Sistemas Informáticos, corresponde la docencia en la asignatura objeto del presente proyecto docente. El personal del departamento está compuesto en la actualidad por aproximadamente un centenar de profesores, a los que se suman un número similar de personal contratado con cargo a proyectos de investigación y una decena de miembros de personal de administración y servicios.

De acuerdo con la web del departamento, 88 profesores están adscritos al área de Lenguajes y Sistemas Informáticos, que es la mayoritaria dentro del Departamento. En cuanto a las categorías, a fecha de septiembre de 2017, el área contaba con 10 Catedráticos de Universidad, 37 Titulares de Universidad, 2 Catedráticos de Escuela Universitaria, 10 Contratados Doctores y 3 Titulares de Escuela Universitaria. El resto eran Profesores Colaboradores, Ayudantes y Profesores Asociados a Tiempo Parcial.

El departamento imparte docencia en la mayoría de los centros de la Universidad de Málaga, aunque su mayor presencia se concentra en la E.T.S.I. Informática, por los contenidos de las titulaciones que se imparten en este centro.

En cuanto a la investigación, existen 5 grupos principales dentro del departamento:

- GISUM, Ingeniería del Software
- (IA)2, Investigación y aplicaciones de inteligencia artificial
- GTCI, Técnicas computacionales en la ingeniería
- SICUMA, Sistemas de información cooperativos
- ICAI, Inteligencia computacional y análisis de imagen

2. Contexto curricular

2.1. El Espacio Europeo de Educación Superior

El 25 de mayo de 1998, los Ministros de Educación de Francia, Alemania, Italia y Reino Unido firmaron en la Universidad de la Sorbona una declaración instando al desarrollo de un Espacio Europeo de Educación Superior (EEES). Este hecho fue el comienzo de un proceso de trascendental importancia para todo el sistema universitario europeo y, por tanto, también para el español. El 19 de junio de 1999, se celebró una nueva conferencia en Bolonia que dio lugar a una Declaración conjunta de los ministros de educación reunidos. En total, fue suscrita por 29 estados europeos (los 15 países que en ese momento integraban la UE, los países del Espacio Europeo de Libre Comercio y varios países del este y centro de Europa). La Declaración de Bolonia sienta las bases para la construcción de un EEES organizado conforme a ciertos

principios (calidad, movilidad, diversidad, competitividad) y orientado hacia la consecución, entre otros, de dos objetivos estratégicos:

- el incremento del empleo en la Unión Europea, y
- la conversión del Sistema Europeo de Formación Superior en un polo de atracción para estudiantes y profesores de otras partes del mundo.

La Declaración de Bolonia tiene carácter político. En consecuencia, enuncia una serie de objetivos y unos instrumentos para lograrlos, pero no fija unos deberes jurídicamente exigibles. Los seis objetivos recogidos son:

- Adopción de un sistema fácilmente legible y comparable de titulaciones, mediante la implantación, entre otras cuestiones, de un suplemento al título.
- Adopción de un sistema basado fundamentalmente en dos ciclos principales.
- Establecimiento de un sistema de créditos homologado.
- Promoción de la cooperación europea para asegurar un nivel de calidad para el desarrollo de criterios y metodologías comparables.
- Promoción de una necesaria dimensión europea en la educación superior, con particular énfasis en el desarrollo curricular.
- Promoción de la movilidad y remoción de obstáculos para el ejercicio libre de la misma por los estudiantes, profesores y personal administrativo de las universidades y otras instituciones de enseñanza superior europea.

Desde el año 2010, el EEES está completamente implementado en España. La integración del sistema universitario español en el EEES ha supuesto:

- La adopción de un sistema académico estructurado en ciclos sucesivos. En la actualidad, existen 3 reales decretos en vigor: el Real Decreto 1393/2007, de 29 de octubre, por el que se establece la ordenación de las enseñanzas universitarias oficiales; el Real Decreto 99/2011, de 28 de enero, por el que se regulan las enseñanzas oficiales de doctorado; y el Real Decreto 43/2015, de 2 de febrero, que modifica los dos anteriores. En el primero se establecía que las enseñanzas universitarias conducentes a la obtención de títulos de carácter oficial y validez en todo el territorio nacional se debían estructurar en tres ciclos, denominados respectivamente Grado, Máster y Doctorado. Los planes de estudio de Grado debían tener 240 créditos, y los de Máster Universitario entre 60 y 120. El R.D. 99/2011 regula la organización de los estudios de doctorado correspondientes al tercer ciclo de las enseñanzas universitarias oficiales

conducentes a la obtención del Título de Doctor o Doctora. Por último, el R.D. 43/2015 modifica los dos anteriores ya que, por una parte, para los estudios de grado, permite a las universidades que los planes de estudio tengan entre 180 y 240 créditos, y para el acceso a los estudios de Doctorado se exige que se haya cursado un grado (o equivalente) y un máster (o equivalente), y un mínimo de 300 créditos ECTS entre ambos.

- El establecimiento de un Sistema Europeo de Créditos Transferible (ECTS, *European Credit Transfer System*), que describe de un modo uniforme en toda Europa la carga de trabajo de cada estudiante en cada materia a través del uso de métodos pedagógicos basados en el aprendizaje activo. Mediante este sistema, se presta atención prioritaria al desarrollo de las destrezas, capacidades y habilidades de los titulados, además de a los contenidos específicos de las materias estudiadas. La estructura de las titulaciones específicas impartidas en un centro mediante este sistema debe estar recogida en las guías docentes según un formato normalizado. La adopción del sistema de créditos europeos está regulada en España por el Ministerio de Educación en el Real Decreto 1125/2003, de 5 de agosto (BOE del 18 de septiembre de 2003), por el que se establece el sistema europeo de créditos y el sistema de calificaciones en las titulaciones de carácter oficial, con validez en todo el territorio nacional. Los créditos ECTS representan los valores numéricos asignados a cada asignatura para describir el trabajo necesario que un estudiante debe realizar para preparar dicha asignatura, teniendo en cuenta que 1 crédito ECTS suponen 25-30 horas de trabajo del estudiante. En los créditos se incluyen clases teóricas, prácticas, seminarios, tutorías, trabajos de campo, horas de estudio, exámenes y otros tipos de evaluaciones. Los créditos ECTS también describen la cantidad de trabajo para cada asignatura en relación con el total necesario para un curso completo. Así, un curso académico son 60 ECTS, que se cursan durante 40 semanas de estudio a tiempo completo, lo que implica una dedicación de 37,5 a 45 horas de trabajo semanales. Más allá de la conversión puramente numérica, el ECTS implica el uso de un sistema de enseñanza similar, basada en el aprendizaje, y una forma normalizada y transparente de recoger la información — las guías docentes— y emitir certificaciones.

- La promoción de la calidad y las buenas prácticas en todas las universidades, para lo cual la calidad en la actividad universitaria será evaluada y acreditada periódicamente mediante criterios objetivos. Todas las agencias nacionales responsables de la evaluación de la calidad se agrupan en una red europea para asegurar su coordinación (ENQA, *European Network of Quality Assurance in Higher Education*). En España, la ANECA es la encargada de evaluar la calidad de las instituciones y centros para impartir

las diferentes titulaciones, así como de expedir las acreditaciones correspondientes. Esta agencia participa en las organizaciones europeas citadas y desarrolla un programa piloto de evaluación y acreditación de titulaciones, de programas de doctorado y de servicios universitarios.

- La redacción de una descripción detallada de cada titulación — con contenidos y formatos uniformes en toda Europa— que haga que las distintas titulaciones de los países sean fácilmente comprensibles y comparables entre sí. Esta descripción es lo que se ha llamado Suplemento Europeo al Título, que contiene información relevante para las agencias de reconocimiento académico y profesional de los distintos países, con el fin de que se pueda determinar la equivalencia en términos de formación, e incluye las competencias profesionales a que acredita el título, la duración y estructura de los estudios y las características e indicadores de calidad del centro y la universidad que expiden el título, además de información sobre el sistema universitario de cada país. Este aspecto fue el primero sobre el EEES aprobado en España, en el Real Decreto 1044/2003, de 1 de agosto (BOE del 11 de septiembre de 2003), por el que se establece el procedimiento para la expedición del Suplemento Europeo al Título por parte de las universidades.

2.2. La estructura de las titulaciones

Los planes de estudios conducentes a titulaciones universitarias oficiales deben ser verificados por el Consejo de Universidades, que los envía a la ANECA para que elabore un informe de evaluación. Además, deben someterse a un procedimiento de evaluación cada 6 años, con el fin de mantener su acreditación. Por último, deben estar autorizados en su implantación por la correspondiente Comunidad Autónoma (artículo 35.2 de la Ley Orgánica 6/2001, modificada por la Ley 4/2007, de Universidades). Los títulos a cuya obtención conduzcan deberán ser inscritos en el Registro de Universidades, Centros y Títulos y acreditados, según las previsiones del R.D. 43/2015.

Las titulaciones oficiales españolas implantadas de acuerdo con el EEES se estructuran en títulos de Grado, de Máster, y de Doctorado.

Títulos de Grado

Las directrices generales comunes que deben cumplir todas las titulaciones oficiales de Grado, según los R.D. 1393/2007, R.D. 99/2011 y R.D. 43/2015, son:

- En los planes de estudios se debe primar la formación básica y generalista y no la especialización del estudiante.

- Los planes de estudios deben tener entre 180 y 240 créditos, que contendrán toda la formación teórica y práctica que el estudiante deba adquirir: aspectos básicos de la rama de conocimiento, materias obligatorias u optativas, seminarios, prácticas externas, trabajos dirigidos, trabajo de fin de Grado u otras actividades formativas.
- Para las titulaciones de Grado que tengan menos de 240 créditos, las Universidades deben arbitrar mecanismos que complementen el número de créditos de Grado con el número de créditos de Máster, de manera que se garantice que la formación del Grado es generalista y los contenidos del Máster se orienten hacia una mayor especialización.
- Deben concluir con la elaboración y defensa de un trabajo de fin de Grado.

Más concretamente, los planes de estudios de las titulaciones oficiales de Grado deben incluir:

- Un número de créditos de formación básica que alcance al menos el 25% del total de los créditos del título.
- Al menos el 60% de los créditos de formación básica serán vinculados a algunas de las materias que se enumeran en el citado decreto, en función de la rama de conocimiento a la que se pretenda adscribir el título, y deberán concretarse en asignaturas de — al menos— 6 créditos cada una. Estas asignaturas deberán ser ofertadas en la primera mitad del plan de estudios.
- Los créditos restantes deberán estar configurados por materias básicas de la misma u otras ramas de conocimiento, o por otras materias siempre que se justifique su carácter básico para la formación inicial del estudiante o su carácter transversal.
- Las prácticas externas, si se incluyen, no podrán suponer más del 25% del total de los créditos del título y deberán ofrecerse preferentemente en la segunda mitad del plan de estudios.
- El trabajo de fin de Grado deberá tener un mínimo de 6 créditos y un máximo del 12,5% del total de créditos del título. Deberá realizarse en la fase final del plan de estudios y estar orientado a la evaluación de competencias asociadas al título.
- La posibilidad de que al menos 6 créditos puedan ser obtenidos por reconocimiento académico en créditos por la participación en actividades universitarias culturales, deportivas, de representación estudiantil, solidarias y de cooperación (aspecto recogido en la LOU).

- Para títulos que habiliten para el ejercicio de actividades profesionales reguladas en España, deberán diseñarse de forma que permitan obtener las competencias necesarias para ejercer esa profesión, según las condiciones establecidas por el Gobierno.

Máster Universitario

Los planes de estudios conducentes a la obtención del título de Máster Universitario:

- Deben ser elaborados por las universidades, verificados de acuerdo con lo establecido en R.D. 43/2015 y, en su elaboración, las universidades primarán la especialización de los estudiantes (R.D. 43/2015).
- Deben tener entre 60 y 120 créditos, e incluir toda la formación teórica y práctica que el estudiante deba adquirir: materias obligatorias, materias optativas, seminarios, prácticas externas, trabajos dirigidos, trabajo de fin de Máster, actividades de evaluación y otras que resulten necesarias según las características propias de cada título.
- Deben concluir con la elaboración y defensa pública de un trabajo de fin de Máster de entre 6 y 30 créditos.
- Cuando se trate de títulos que habiliten para el ejercicio de actividades profesionales reguladas en España, deben diseñarse de forma que permitan obtener las competencias necesarias para ejercer esa profesión, según las condiciones establecidas por el Gobierno.

Programas de Doctorado

Las directrices para los estudios de Doctorado se establecen en el R.D. 99/2011. El R.D. 43/2015 solo modifica aquel en las condiciones que establece para el acceso a un programa de doctorado. Por otra parte, el doctorado se establece definitivamente como tercer ciclo de los estudios europeos, diferenciado del máster a partir del Comunicado de la Conferencia de Bergen (2005), en la que se utilizó, como base del comunicado, el proyecto piloto *Doctoral Programmes for the European Knowledge Society*, promovido por la *European University Association*. Según se indica en R.D. 99/2011, se considera que en este tercer ciclo los participantes en programas de doctorado no son solo estudiantes, sino investigadores en formación. Con ello, se enlaza a partir de ese momento la formación doctoral, la carrera investigadora y la transmisión del conocimiento a la sociedad.

En la formación doctoral se debe promover la colaboración de otros organismos, entidades e instituciones implicadas en la I+D+i tanto nacional como internacional. En el citado R.D. se

incluye la creación de las Escuelas de Doctorado, que deben tener un papel esencial en estas enseñanzas.

Los estudios de doctorado se organizan a través de programas, según los estatutos de cada universidad y respetando lo establecido en el R.D. 99/2011. Estos estudios finalizan con la elaboración y defensa de una tesis doctoral que incorpore resultados originales de investigación. Su duración es de 3 años para estudiantes a tiempo completo y 5 para los de tiempo parcial (prorrogables por la comisión académica responsable del programa).

En estos estudios no se requiere la estructuración en créditos ECTS. Se establecen los criterios para la supervisión y seguimiento de los doctorandos durante su periodo de estudios y la obligación de tener un documento de actividades personalizadas que realiza cada doctorado para su revisión por el tutor y director de la tesis.

2.3. Normativa propia de la Universidad de Málaga

Se recogen en esta sección dos aspectos de la normativa de la Universidad de Málaga que han de ser tenidos en cuenta para la elaboración de cualquier proyecto docente de una asignatura que vaya a ser impartida en la misma: las guías docentes y la normativa referida a la evaluación de los alumnos.

Las guías docentes

La guía docente de una asignatura es un documento público dirigido fundamentalmente a los estudiantes y escrito con un lenguaje muy claro, en el que se concreta, para un determinado curso académico, la planificación docente de una asignatura y toda la información necesaria para su seguimiento. Constituye el resultado del compromiso del equipo docente que la va a impartir y del departamento al que se encuentra adscrita, avalado por la universidad a través de los órganos con competencia en la aprobación de la programación docente.

Según el Plan de Ordenación Docente de la Universidad de Málaga, las guías docentes tienen un papel muy importante en diversos aspectos relacionados con los procesos de enseñanza y aprendizaje, con la coordinación docente, con la acreditación de la titulación y con la transparencia de información sobre la docencia:

- Su elaboración proporciona la oportunidad y el método para revisar y mejorar la calidad de los procesos de enseñanza-aprendizaje, definiendo con precisión qué competencias debe lograr el estudiante y cuáles serán los criterios y niveles de exigencia que se aplicarán, así como el diseño de actividades formativas que fomenten la adquisición de las competencias.

- Fomentan la autonomía e implicación del alumnado. Los estudiantes necesitan saber hacia dónde les conduce cada asignatura y la titulación en su conjunto para percibir el sentido y la relevancia de su esfuerzo.
- Implican un compromiso de trabajo conjunto por parte del equipo docente que la elabora, que se plasma en una planificación docente común, entendiendo que es la mejor formación que puede ofrecer a los estudiantes que van a cursar dicha asignatura.
- Hacen posible la coordinación de la titulación. Para que se pueda lograr una adecuada integración de todas las asignaturas que conforman una titulación y una correcta coordinación entre las personas implicadas en las mismas, es necesario saber qué se hace y qué se exige en cada una de ellas.
- Contribuyen a la acreditación de la titulación. El título, como organización, necesita establecer con precisión qué competencias logra el estudiante en cada una de sus partes integrantes y si los criterios y niveles de exigencia que se aplican son los adecuados en cada caso.
- Concretan el compromiso de la institución universitaria con los estudiantes que se han matriculado en una titulación. En este sentido, la guía docente representa el documento que permite al estudiante realizar un seguimiento del cumplimiento de la planificación docente de una asignatura.
- Constituyen un instrumento con información transparente, fácilmente comprensible y comparable, entre las diferentes universidades en el camino hacia la convergencia en un EEES.

Los Estatutos de la Universidad de Málaga (BOJA núm. 108, de 9 de junio de 2003) en su artículo 134, punto b, establecen que la programación docente de cada asignatura contendrá al menos: el temario, la metodología pedagógica y el sistema de evaluación del rendimiento académico de los alumnos, fijando el tipo de pruebas, su número, los criterios para su corrección y los componentes que se tendrán en cuenta para la calificación final del alumnado.

Por su parte, las normas reguladoras de la realización de las pruebas de evaluación del rendimiento académico de los estudiantes de enseñanzas oficiales de primer y segundo ciclo (aprobadas en Consejo de Gobierno de 18 de diciembre de 2009) recogen que la programación docente, de acuerdo con las previsiones del artículo 134 de los Estatutos de la Universidad de Málaga, deberá contener:

- Los objetivos docentes, los contenidos, la metodología y el sistema de evaluación de las competencias y conocimientos de cada asignatura.

- El programa y actividades de cada asignatura, con su bibliografía básica y complementaria. El departamento responsable asegurará un único programa por asignatura cuando ésta se divida en más de un grupo docente.
- El profesorado previsto para la docencia, así como los horarios de las enseñanzas.
- El cronograma de aplicación de los sistemas de evaluación en cada una de las convocatorias previstas.
- El porcentaje de éxito en la asignatura, de los tres últimos cursos académicos.

La información de la planificación de cada asignatura se debe incluir en la guía docente, según el modelo de guía docente propuesto en el Plan de Ordenación Docente (POD) de la Universidad de Málaga. Según se recoge en el POD. la guía docente de una asignatura es un documento público dirigido, fundamentalmente, a los estudiantes y escrito con un lenguaje claro y conciso, en el que se describen, para cada curso académico, los siguientes aspectos:

- Descripción de la asignatura
- Equipo docente
- Recomendaciones y orientaciones
- Contexto de la asignatura
- Competencias
- Contenidos
- Actividades formativas
- Resultados de aprendizaje/criterios de evaluación
- Procedimiento de evaluación
- Bibliografía y otros recursos
- Distribución del trabajo del estudiante
- Cronograma

La normativa de evaluación en la Universidad de Málaga

La regulación del procedimiento a seguir en la Universidad de Málaga para la valoración del progreso y los resultados del aprendizaje de los estudiantes, con carácter general, se contempla en el artículo 134 de sus Estatutos, aprobados por Decreto de la Junta de Andalucía número 145/2003, de 3 de junio (BOJA del 9 de junio).

De acuerdo con lo establecido en el mencionado artículo, para cada curso académico, y con antelación suficiente al inicio del correspondiente período lectivo, las Juntas de Centro, a partir de la información facilitada por los correspondientes Departamentos, aprobarán el programa académico de las enseñanzas correspondientes a las titulaciones oficiales que se imparten en el

respectivo Centro. Dicho programa deberá incluir, entre otros extremos, la programación docente de cada una de las correspondientes asignaturas, y ésta, a su vez, deberá incorporar el sistema de evaluación del rendimiento académico de los alumnos, fijando el tipo de pruebas, su número, los criterios para su corrección y los componentes que se tendrán en cuenta para la calificación final del estudiante.

El mencionado sistema de evaluación debe, a su vez, tener presente lo preceptuado en el artículo 124 de los citados Estatutos, que establece el derecho de los mencionados estudiantes a presentarse a dos convocatorias ordinarias de examen por curso académico. Además del citado procedimiento de carácter general, consecuencia del régimen jurídico vigente en la materia, la valoración del progreso y los resultados del aprendizaje de los estudiantes se contempla también en el procedimiento PE03 (Medición, Análisis y Mejora Continua) del Sistema de Garantía de Calidad, con la finalidad de lograr la mejora de la calidad de la enseñanza.

De acuerdo con el Informe sobre Innovación de la Docencia en las Universidades Andaluzas (CIDUA), la valoración del progreso y los resultados del aprendizaje de los estudiantes se llevará teniéndose presente que es preciso considerar la evaluación como una ocasión para conocer la calidad de los procesos de enseñanza-aprendizaje y una oportunidad para su reformulación y mejora.

Se impone la necesidad de ampliar el concepto de evaluación del rendimiento para que abarque los diferentes componentes de las competencias personales y profesionales que se propone desarrollar la enseñanza universitaria: conocimientos, habilidades, actitudes y comportamientos. La pretensión central del modelo de evaluación que propone la Universidad de Málaga es que el estudiante en todo momento tenga conciencia de su proceso de aprendizaje, comprenda lo que aprende, sepa aplicarlo y entienda el sentido y la utilidad social y profesional de los aprendizajes que realiza.

Los apoyos metodológicos fundamentales del proyecto docente que orientan el modelo marco propuesto descansan en la combinación del trabajo individual, las explicaciones del docente, la experimentación en la práctica, la interacción y el trabajo cooperativo entre iguales y la comunicación con el tutor.

En definitiva, se trata de transformar el modelo convencional, de transmisión oral de conocimientos, toma de apuntes y reproducción de lo transmitido en pruebas y exámenes, en un modelo que reafirma la naturaleza tutorial de la función docente universitaria, que atiende a las peculiaridades del aprendizaje profesional y académico de cada estudiante.

2.4. Currículos de la ACM/IEEE

Desde 1968, ha ido apareciendo una serie de informes técnicos elaborados por la *Association for Computer Machinery* (ACM), el *Institute of Electrical and Electronics Engineers* (IEEE), la Unesco y la Universidad Carnegie-Mellon, con el fin de desarrollar un conjunto de guías para la confección de currículos dentro de las disciplinas de las Ciencias de la Computación.

En 1988 se coordinan los grupos de ACM y la *Computer Society* de IEEE con el fin de agrupar y actualizar lo que hasta entonces habían constituido recomendaciones curriculares independientes — ACM (1965), COSINE (1967), ACM (1968), ACM (1979), y *Board* (1983)—. Posteriores revisiones y modificaciones por parte de educadores y expertos en el área llevaron a un trabajo final, el *Computing Curricula 1991 ACM*. Este trabajo se ha ido actualizando y en el año 2005 se elaboraron los currículos para 5 subdisciplinas de la computación: *Computer Science*, *Computer Engineering*, *Information Systems*, *Information Technology* y *Software Engineering*.

En la disciplina de *Software Engineering* se establecen guías para el desarrollo de cursos introductorios a la Ingeniería del Software. La versión más reciente de esta guía es de 2020 [1]. En ella, se establecen las siguientes competencias básicas para la disciplina, que incluyen:

- **Requisitos software.** Incluyen la (1) identificación y documentación de requisitos software, mediante la aplicación de métodos de elicitación de requisitos adecuadas, trabajando en sesiones con stakeholders; (2) análisis de requistos, estableciendo su coherencia, completitud, y factibilidad, y recomendar mejoras en la documentación de requisitos (3) especificación de requisitos software utilizando formatos estándar y lenguajes seleccionados para el proyecto, siendo capaz de describir los requisitos de forma clara a personas no expertas tales como usuarios finales; (3) verificación y validación de requisitos usando técnicas estándar, incluyendo inspección, modelado, prototipado, y casos de test; (5) seguir los procedimientos de gestión de procesos y producto que hayan sido identificados para el proyecto.

- **Diseño del software.** Incluyen la (1) presentación a las personas encargadas de la toma de decisiones en el negocio de requisitos arquitectónicamente relevantes a partir de la especificación de requisitos del software; (2) evaluación y comparación de tradeoffs de distintas alternativas de diseño para satisfacer los requisitos funcionales y no funcionales, así como escritura de propuestas breves que resuman las conclusiones principales para el cliente; (3) producción de un diseño de alto nivel de subsistemas específicos que sea presentable a un público no experto en computación, sonsiderando patrones arquitectónicos y de diseño; (4) producir diseños detallados para un cliente para subsistemas específicos mediante el uso de principios de diseño y consideración

de aspectos interrelacionados para satisfacer requisitos funcionales y no funcionales; (5) evaluación de consideraciones para las pruebas de software relacionadas con los atributos de calidad en el diseño de subsistemas y módulos para un desarrollador/productor; (6) crear documentos de diseño del software que comuniquen de forma efectiva a los consumidores del diseño tales como analistas, desarrolladores, o mantenedores del software.

- **Construcción del software.** Incluyen el (1) diseño e implementación de APIs utilizando lenguajes orientados a objetos y librerías extendidas, incluyendo parametrización y aspectos genéricos de pequeños proyectos; (2) evaluación de un sistema software respecto a prácticas modernas del software tales como la programación defensiva, y la gestión de errores y excepciones, tolerancia a fallos, de tal modo que permitan la inclusión de mecanismos de gestión en tiempo de ejecución dentro de grandes proyectos; (3) desarrollo de un sistema basado en la nube que incorpore primitivas de concurrencia para proyectos de media escala, así como el análisis de rendimiento y afinado del sistema.

- **Pruebas del software.** Incluye la (1) prueba y análisis de integración de componentes software mediante técnicas de caja negra y casos de uso en colaboración con clientes; (2) conducción de pruebas de regresión de componentes software para un cliente que consideren perfiles operacionales y atributos de calidad específicos a una aplicación a partir de datos empíricos y uso planeado; (3) realización de pruebas utilizando herramientas adecuadas, centrándose en los atributos de calidad deseables especificados por el equipo de control de calidad y el cliente; (4) planear y realizar el diseño de casos de prueba para una organización utilizando técnicas de *clear-box* y *black-box* para medir las métricas de calidad en términos de cobertura y rendimiento.

- **Sostenimiento del software.** Incluye la (1) descripción de criterios para realizar la transición del software a un estatus de sostenimiento y asistencia en la identificación de estándares operacionales aplicables; (2) identificación de necesidades de personal de soporte operacional, documentación y formación necesarias para facilitar el desarrollo de la documentación de soporte y formación necesarias para la transición; (3) determinación del impacto de los cambios en el software en el entorno de operación; (4) descripción de los elementos de las actividades de soporte del software en el entorno de operación; (5) realización de actividades de soporte del software, e interacción efectiva con otro personal de soporte; (6) asistir en la implementación de planes y procesos de mantenimiento del software, así como realización de cambios al software para satisfacer necesidades y peticiones de mantenimiento.

Si bien es cierto que el número de horas para el aprendizaje es limitado, el objetivo de una asignatura introductoria a la ingeniería del software no es hacer que los alumnos adquieran todas y cada una de las competencias mencionadas anteriormente, sino dar una base sólida en el conocimiento de las técnicas del desarrollo del software, familiarizándoles con los cinco bloques básicos mencionados anteriormente.

3. Grado en Ingeniería del Software de la Universidad de Málaga

El centro responsable de la titulación de Graduado/a en Ingeniería del Software de la Universidad de Málaga es la Escuela Técnica Superior de Ingeniería Informática. Los principales datos del título son:

Denominación del título: Graduado/a en Ingeniería del Software

Centro Responsable: Escuela Técnica Superior de Ingeniería Informática

Créditos: 240 ECTS – 4 año(s)

Rama: Ingeniería y Arquitectura

Tipo de enseñanza: Presencial

Profesión regulada: no

Publicación en BOE: 20-10-2011

1º Curso de implantación: 2010

Número de plazas de nuevo ingreso: 65

Lenguas utilizadas: Castellano, inglés

Estado: Implantado

En cuanto a los resultados, los datos publicados por el centro son los siguientes:

	Curso 16/17	**Curso 17/18**
Tasa de abandono	18.18%	15.63%
Tasa de eficiencia	84.15%	89.85%
Tasa de rendimiento	68.23%	66.95%
Tasa de éxito	84.56%	84.30%

Los 240 créditos ECTS del Plan de Estudios se distribuyen por tipo de materia de la siguiente manera:

- Formación básica: 60 ECTS
- Formación obligatoria: 138 ECTS
- Formación optativa: 30 ECTS
- Prácticas externas: 0 ECTS
- Trabajo Fin de Grado: 12 ECTS

3.1. Competencias

Competencias genéricas

CG01. Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.

CG02. Capacidad para dirigir las actividades objeto de los proyectos del ámbito de la informática de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG03. Capacidad para diseñar, desarrollar, evaluar y asegurar la accesibilidad, ergonomía, usabilidad y seguridad de los sistemas, servicios y aplicaciones informáticas, así como de la información que gestionan.

CG04. Capacidad para definir, evaluar y seleccionar plataformas hardware y software para el desarrollo y la ejecución de sistemas, servicios y aplicaciones informáticas, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG05. Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG06. Capacidad para concebir y desarrollar sistemas o arquitecturas informáticas centralizadas o distribuidas integrando hardware, software y redes de acuerdo con los

conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG07. Capacidad para conocer, comprender y aplicar la legislación necesaria durante el desarrollo de la profesión de Ingeniero Técnico en Informática y manejar especificaciones, reglamentos y normas de obligado cumplimiento.

CG08. Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

CG09. Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática.

CG10. Conocimientos para la realización de mediciones, cálculos, valoraciones, tasaciones, peritaciones, estudios, informes, planificación de tareas y otros trabajos análogos de informática, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG11. Capacidad para analizar y valorar el impacto social y medioambiental de las soluciones técnicas, comprendiendo la responsabilidad ética y profesional de la actividad del Ingeniero Técnico en Informática.

CG12. Conocimiento y aplicación de elementos básicos de economía y de gestión de recursos humanos, organización y planificación de proyectos, así como la legislación, regulación y normalización en el ámbito de los proyectos informáticos, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG13. Capacidad de expresión oral y escrita en un segundo idioma (inglés).

Competencias de Tecnología Específica

CE-IS-01. Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la Ingeniería del Software.

CE-IS-02. Capacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la

búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones.

CE-IS-03. Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles.

CE-IS-04. Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.

CE-IS-05. Capacidad de identificar, evaluar y gestionar los riesgos potenciales asociados que pudieran presentarse.

CE-IS-06. Capacidad para diseñar soluciones apropiadas en uno o más dominios de aplicación utilizando métodos de la ingeniería del software que integren aspectos éticos, sociales, legales y económicos.

CE-IS-07. Capacidad para manejar herramientas de modelado y diseño del software que permitan la construcción, operación y mantenimiento de aplicaciones software de forma sistemática, medible y rigurosa.

CE-IS-08. Capacidad para seleccionar y utilizar las metodologías de desarrollo y tecnologías de implementación más adecuadas a los requisitos de los clientes.

CE-IS-09. Capacidad para manejar herramientas de desarrollo del software que permitan la construcción, operación y mantenimiento de aplicaciones software de forma sistemática, medible y rigurosa.

CE-IS-10. Capacidad para analizar formalmente y razonar rigurosamente sobre la corrección y las propiedades de los sistemas construidos.

CE-IS-11. Capacidad para desenvolverse en un entorno laboral, asimilando el funcionamiento y organización de una empresa, y sabiendo aplicar los conocimientos adquiridos en un entorno empresarial en el contexto de algunas de las tecnologías específicas desarrollada en el curriculum.

Formación Básica

CB01. Capacidad para la resolución de los problemas matemáticos que puedan plantearse en la ingeniería. Aptitud para aplicar los conocimientos sobre: álgebra lineal; cálculo diferencial e integral; métodos numéricos; algorítmica numérica; estadística y optimización.

CB02. Comprensión y dominio de los conceptos básicos de campos y ondas y electromagnetismo, teoría de circuitos eléctricos, circuitos electrónicos, principio físico de los semiconductores y familias lógicas, dispositivos electrónicos y fotónicos, y su aplicación para la resolución de problemas propios de la ingeniería.

CB03. Capacidad para comprender y dominar los conceptos básicos de matemática discreta, lógica, algorítmica y complejidad computacional, y su aplicación para la resolución de problemas propios de la ingeniería.

CB04. Conocimientos básicos sobre el uso y programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación en ingeniería.

CB05. Conocimiento de la estructura, organización, funcionamiento e interconexión de los sistemas informáticos, los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la ingeniería.

CB06. Conocimiento adecuado del concepto de empresa, marco institucional y jurídico de la empresa. Organización y gestión de empresas. Competencias específicas.

Formación Común

CC01. Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.

CC02. Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social.

CC03. Capacidad para comprender la importancia de la negociación, los hábitos de trabajo efectivos, el liderazgo y las habilidades de comunicación en todos los entornos de desarrollo de software.

CC04. Capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes.

CC05. Conocimiento, administración y mantenimiento sistemas, servicios y aplicaciones informáticas.

CC06. Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos.

CC07. Conocimiento, diseño y utilización de forma eficiente los tipos y estructuras de datos más adecuados a la resolución de un problema.

CC08. Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.

CC09. Capacidad de conocer, comprender y evaluar la estructura y arquitectura de los computadores, así como los componentes básicos que los conforman.

CC10. Conocimiento de las características, funcionalidades y estructura de los Sistemas Operativos y diseñar e implementar aplicaciones basadas en sus servicios.

CC11. Conocimiento y aplicación de las características, funcionalidades y estructura de los Sistemas Distribuidos, las Redes de Computadores e Internet y diseñar e implementar aplicaciones basadas en ellas.

CC12. Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.

CC13. Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.

CC14. Conocimiento y aplicación de los principios fundamentales y técnicas básicas de la programación paralela, concurrente, distribuida y de tiempo real.

CC15. Conocimiento y aplicación de los principios fundamentales y técnicas básicas de los sistemas inteligentes y su aplicación práctica.

CC16. Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.

CC17. Capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas.

CC18. Conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional.

CC19. Capacidad para comprender y dominar los conceptos básicos de autómatas y lenguajes formales, y su aplicación para la resolución de problemas propios de la informática.

CC20. Capacidad para comprender y dominar los conceptos relativos a la calculabilidad, decidibilidad y enumerabilidad, y su relevancia en los problemas propios de la informática.

3.2. Organización de las materias

El plan de estudios se organiza en las siguientes materias y asignaturas:

ESTRUCTURA DE LAS ENSEÑANZAS POR MÓDULOS Y MATERIAS

Módulo de Formación Básica (60 Créditos)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
Matemáticas (18 créditos)	*Estructuras Algebraicas para la Computación*	*6*	*BA*
	Matemática Discreta	*6*	*BA*
	Cálculo para la Computación	*6*	*BA*
Física (12 créditos)	*Fundamentos Físicos de la Informática*	*6*	*BA*
	Fundamentos de Electrónica	*6*	*BA*
Informática (18 créditos)	*Fundamentos de la Programación*	*6*	*BA*
	Programación Orientada a Objetos	*6*	*BA*
	Tecnología de Computadores	*6*	*BA*
Estadística (6 créditos)	*Métodos Estadísticos para la Computación*	*6*	*BA*
Empresa (6 créditos)	*Organización Empresarial*	*6*	*BA*

Módulo de Formación Común (60 Créditos)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
	Estructuras de Datos	*6*	*OB*
	Análisis y Diseño de Algoritmos	*6*	*OB*

Programación de Computadores (18 créditos)	*Programación de Sistemas y Concurrencia*	*6*	*OB*
Ingeniería del Software, Sistemas de Información y Sistemas Inteligentes (18 créditos)	*Bases de Datos*	*6*	*OB*
	Sistemas Inteligentes	*6*	*OB*
	Introducción a la Ingeniería del Software	*6*	*OB*
Sistemas Operativos, Sistemas Distribuidos y Redes y Arquitectura de Computadores (18 créditos)	*Redes y Sistemas Distribuidos*	*6*	*OB*
	Sistemas Operativos	*6*	*OB*
	Estructura de Computadores	*6*	*OB*
Elaboración de Proyectos Informáticos (6 créditos)	*Proyectos y Legislación*	*6*	*OB*

Módulo de Fundamentos de la Computación (6 Créditos)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
Teoría de Autómatas y Lenguajes Formales (6 créditos)	*Teoría de Autómatas y Lenguajes Formales*	*6*	*OB*

Módulo de Proyecto Fin de Grado (12 Créditos)

Materias	**Asignatura**	**Créditos ECTS**	**Carácter**
Trabajo Fin de Grado (12 créditos)	*Trabajo Fin de Grado*	*12*	*TFG*

Módulo de Complementos de la Ingeniería Informática (210 Créditos)(*)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
	Ampliación de Física	*6*	*OP*

Complementos de Electrónica y Física (24 créditos)	*Electrónica para Domótica*	*6*	*OP*
	Herramientas del Diseño Electrónico	*6*	*OP*
	Electrónica Digital	*6*	*OP*
Automática (18 créditos)	*Modelado y Simulación de Sistemas*	*6*	*OP*
	Sistemas de Automatización	*6*	*OP*
	Programación de Robots	*6*	*OP*
Complementos de Percepción y Razonamiento (18 créditos)	*Inteligencia Artificial para Juegos*	*6*	*OP*
	Procesamiento de Imágenes y Video	*6*	*OP*
	Visión por Computador	*6*	*OP*
Software Multimedia (18 créditos)	*Programación Gráfica 3D*	*6*	*OP*
	Programación de Videojuegos	*6*	*OP*
	Servicios Multimedia	*6*	*OP*
Complementos de Arquitectura de Computadores (12 créditos)	*Arquitecturas Virtuales*	*6*	*OP*
	Arquitecturas Cluster	*6*	*OP*
Complementos de Sistemas Distribuidos (18 créditos)	*Desarrollo de Software Crítico*	*6*	*OP*
	Ingeniería de Protocolos	*6*	*OP*
	Redes Inalámbricas	*6*	*OP*
Fundamentos y Complementos Transversales (30 créditos)	*Fundamentos de Economía y Política Económica*	*6*	*OP*
	Laboratorio de Computación Científica	*6*	*OP*
	Teoría de Dominios y Modelos Computacionales	*6*	*OP*
	Teoría de la Señal	*6*	*OP*
	Teoría de la Información y la Codificación	*6*	*OP*

Calidad de Software (6 créditos)	*Calidad de Software*	*6*	*OP*
Desarrollo Dirigido por Modelos (6 créditos)	*Desarrollo de Software Dirigido por Modelos*	*6*	*OP*
Cognición y Comunicación en Ingeniería del Software (6 créditos)	*Cognición y Comunicación en Ingeniería del Software*	*6*	*OP*
Arquitecturas Especializadas (6 créditos)	*Arquitecturas Especializadas*	*6*	*OP*
Sensores y Actuadores (6 créditos)	*Sensores y Actuadores*	*6*	*OP*
Microelectrónica (6 créditos)	*Implementación Electrónica de Procesadores*	*6*	*OP*
Diseño de Equipos y Sistemas Electrónicos (6 créditos)	*Diseño de Equipos y Sistemas Electrónicos*	*6*	*OP*
Inteligencia Computacional (6 créditos)	*Inteligencia Computacional*	*6*	*OP*
Lógica e Informática (6 créditos)	*Lógica e Informática*	*6*	*OP*
Métodos Matemáticos para la Gestión Inteligente de la Información (6 créditos)	*Gestión Inteligente de la Información*	*6*	*OP*
Programación Declarativa (6 créditos)	*Programación Declarativa*	*6*	*OP*

Sistemas de Información para la Industria (6 créditos)	*Sistemas de Información para la Industria*	*6*	*OP*

Módulo de Prácticas Externas (12 Créditos)(*)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
Prácticas Externas (12 créditos)	*Prácticas Externas*	*12*	*OP*

Módulo de Ingeniería del Software I (48 Créditos)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
Proceso del Desarrollo del Software (24 créditos)	*Ingeniería de Requisitos*	*6*	*OB*
	Modelado y Diseño del Software	*6*	*OB*
	Gestión de Proyectos Software	*6*	*OB*
	Mantenimiento y Pruebas del Software	*6*	*OB*
Tecnologías de Desarrollo (18 créditos)	*Gestión de la Información*	*6*	*OB*
	Ingeniería Web	*6*	*OB*
	Seguridad en Servicios y Aplicaciones	*6*	*OB*
Técnicas Computacionales (6 créditos)	*Técnicas Computacionales para la Ingeniería del Software*	*6*	*OB*

Módulo de Ingeniería del Software II (24 Créditos)

Materias	**Asignaturas**	**Créditos ECTS**	**Carácter**
Interfaces de Usuario (6 créditos)	*Interfaces del Usuario*	*6*	*OB*

Tecnologías de Aplicaciones Web (6 créditos)	*Tecnologías de Aplicaciones Web*	*6*	*OB*
Métodos Formales para la Ingeniería del Software (6 créditos)	*Métodos Formales para la Ingeniería del software*	*6*	*OB*
Software para Sistemas Empotrados y Dispositivos Móviles (6 créditos)	*Software para Sistemas Empotrados y Dispositivos Móviles*	*6*	*OB*

(*) En el caso de optar por la realización de la asignatura "Prácticas Externas" (12 créditos), los estudiantes habrán de elegir, además, tres asignaturas optativas de 6 créditos de entre las relacionadas.

3.3. Organización temporal

Las asignaturas relacionadas con anterioridad se organizan temporalmente de la siguiente manera:

ORGANIZACIÓN TEMPORAL DEL PLAN DE ESTUDIOS

PRIMER CURSO

Asignaturas	**Semestre**	**Carácter**	**ECTS**
Cálculo para la Computación	*1*	*BA*	*6*
Fundamentos de Electrónica	*1*	*BA*	*6*
Fundamentos de la Programación	*1*	*BA*	*6*
Fundamentos Físicos de la Informática	*1*	*BA*	*6*
Matemática Discreta	*1*	*BA*	*6*
Estructuras Algebraicas para la Computación	*2*	*BA*	*6*
Métodos Estadísticos para la Computación	*2*	*BA*	*6*
Organización Empresarial	*2*	*BA*	*6*

Programación Orientada a Objetos	*2*	*BA*	*6*
Tecnología de Computadores	*2*	*BA*	*6*

SEGUNDO CURSO

Asignaturas	**Semestre**	**Carácter**	**ECTS**
Análisis y Diseño de Algoritmos	*1*	*OB*	*6*
Bases de Datos	*1*	*OB*	*6*
Estructura de Computadores	*1*	*OB*	*6*
Estructuras de Datos	*1*	*OB*	*6*
Teoría de Autómatas y Lenguajes Formales	*1*	*OB*	*6*
Introducción a la Ingeniería del Software	***2***	***OB***	***6***
Programación de Sistemas y Concurrencia	*2*	*OB*	*6*
Redes y Sistemas Distribuidos	*2*	*OB*	*6*
Sistemas Inteligentes	*2*	*OB*	*6*
Sistemas Operativos	*2*	*OB*	*6*

TERCER CURSO

Asignaturas	**Semestre**	**Carácter**	**ECTS**
Gestión de la Información	*1*	*OB*	*6*
Ingeniería de Requisitos	*1*	*OB*	*6*
Modelado y Diseño del Software	*1*	*OB*	*6*
Técnicas Computacionales para la Ingeniería del Software	*1*	*OB*	*6*
Optativa I (ver relación de asignaturas optativas)	*1*	*OP*	*6*
Interfaces del Usuario	*2*	*OB*	*6*
Mantenimiento y Pruebas del Software	*2*	*OB*	*6*
Seguridad en Servicios y Aplicaciones	*2*	*OB*	*6*

Tecnología de Aplicaciones Web	*2*	*OB*	*6*
Optativa II (ver relación de asignaturas optativas)	*2*	*OP*	*6*

CUARTO CURSO

Asignaturas	**Semestre**	**Carácter**	**ECTS**
Gestión de Proyectos Software	*1*	*OB*	*6*
Ingeniería Web	*1*	*OB*	*6*
Métodos Formales para la Ingeniería del Software	*1*	*OB*	*6*
Software para Sistemas Empotrados y Dispositivos Móviles	*1*	*OB*	*6*
Optativa III (ver relación de asignaturas optativas)	*1*	*OP*	*6*
Proyectos y Legislación	*2*	*OB*	*6*
Prácticas Externas u Optativas IV y V (ver relación de asignaturas optativas)	*2*	*OP*	*12*
Trabajo Fin de Grado	*2*	*TFG*	*12*

Relación de Asignaturas Optativas

Asignaturas	**ECTS**
Ampliación de Física	*6*
Electrónica para Domótica	*6*
Herramientas del Diseño Electrónico	*6*
Electrónica Digital	*6*
Modelado y Simulación de Sistemas	*6*
Sistemas de Automatización	*6*
Programación de Robots	*6*
Inteligencia Artificial para Juegos	*6*
Procesamiento de Imágenes y Video	*6*

Visión por Computador	*6*
Programación Gráfica 3D	*6*
Programación de Videojuegos	*6*
Servicios Multimedia	*6*
Arquitecturas Virtuales	*6*
Arquitecturas Cluster	*6*
Desarrollo de Software Crítico	*6*
Ingeniería de Protocolos	*6*
Redes Inalámbricas	*6*
Fundamentos de Economía y Política Económica	*6*
Laboratorio de Computación Científica	*6*
Teoría de Dominios y Modelos Computacionales	*6*
Teoría de la Señal	*6*
Teoría de la Información y la Codificación	*6*
Calidad de Software	*6*
Desarrollo de Software Dirigido por Modelos	*6*
Cognición y Comunicación en Ingeniería del Software	*6*
Arquitecturas Especializadas	*6*
Sensores y Actuadores	*6*
Implementación Electrónica de Procesadores	*6*
Diseño de Equipos y Sistemas Electrónicos	*6*
Inteligencia Computacional	*6*
Lógica e Informática	*6*
Gestión Inteligente de la Información	*6*
Programación Declarativa	*6*

Sistemas de Información para la Industria	*6*
Prácticas Externas	*12*

En el caso de optar por la realización de la asignatura "Prácticas Externas" (12 créditos), los estudiantes habrán de elegir, además, tres asignaturas optativas de 6 créditos de entre las relacionadas. En caso de optar por la no realización de la asignatura "Prácticas Externas" (12 créditos), los estudiantes habrán de elegir cinco asignaturas optativas de 6 créditos de entre las relacionadas.

PARTE II. OBJETIVOS, METODOLOGÍA Y EVALUACIÓN

Al profesor, como responsable directo e inmediato de la docencia en la Universidad, le compete la tarea de diseñar y planificar las intenciones y acciones que constituyen las asignaturas a impartir.

La planificación significa inclinarse por una racionalización de la docencia en vez de una improvisación [4]. Esto nos lleva a la necesidad de hacer explícitos los elementos que constituyen el currículo, y que son:

- *Objetivos previos,* es decir, los logros a alcanzar mediante la docencia.
- *Contenidos*, tradicional núcleo central de la planificación curricular, los cuales deben ser congruentes con los objetivos.
- *Actividades didácticas,* es decir, las tareas que docente y discente desarrollan en el proceso de la enseñanza.
- *Evaluación*, mediante la cual se obtendrá una medición y valoración de los resultados, y que proporcionará la necesaria retroalimentación con la que es posible revisar permanentemente la planificación docente.

En el capítulo anterior se ha presentado el contexto en el que se pretende desarrollar este proyecto docente para la asignatura de introducción a la ingeniería del software. Se desarrollan en este capítulo algunos conceptos generales relativos a los objetivos educativos, la metodología docente y la evaluación, que son necesarios para llevar a cabo la planificación de una asignatura. Por último, se presentan las recomendaciones para la docencia universitaria elaboradas en la Comunidad de Andalucía. Todo ello servirá de base para la elaboración del programa de la asignatura objeto de esta memoria.

1. Los objetivos educativos

Es importante distinguir entre fines y objetivos de la asignatura. Los objetivos son la especificación de los resultados pretendidos del aprendizaje del alumno. Los fines, en cambio, son pautas o criterios generales que clarifican y orientan la actividad educativa, y se caracterizan por su amplitud y poca concreción [5]. Debido a estas características, los fines deben traducirse en logros más inmediatos, cercanos a situaciones concretas; dichos logros se especifican a través de los objetivos. Las características que debe tener un objetivo educativo pueden resumirse en: ser claro y preciso, dirigido al alumno, realista y alcanzable, temporalizado y comunicado al alumno [5].

Según su amplitud, los objetivos pueden clasificarse en generales y específicos. Los objetivos generales expresan el aprendizaje final que se pretende obtener, mientras que los específicos expresan los aprendizajes o cambios educativos intermedios para lograr dicho aprendizaje final.

Una clasificación de los objetivos muy empleada es la Taxonomía de Bloom [5]. Según esta taxonomía, los objetivos se clasifican en cognitivos y afectivos, en función del campo de aprendizaje en el que se aplican. Estos, a su vez, se subdividen en niveles:

1. **Dominio cognitivo.** Los objetivos pertenecientes a este dominio son los que se refieren a los aprendizajes relativos a los saberes o conocimientos. Se dividen en seis categorías, que implican sucesivamente destrezas cognitivas cada vez más complejas:
    - Conocimiento: mera adquisición de conocimientos, memorización y evocación de conceptos.
    - Comprensión: asimilación de la información recibida, sin tener que relacionarla con otra información.
    - Aplicación: utilización de conceptos previamente conocidos en situaciones concretas.
    - Análisis: descomposición de un todo en sus partes, viendo las interrelaciones.
    - Síntesis: reunión por parte del sujeto de elementos o partes con el fin de formar un todo que es nuevo para él. Esta categoría es la que más fomenta la actividad creativa.
    - Evaluación: formulación de juicios cualitativos y cuantitativos, de manera que el sujeto, en función de unos criterios dados, valora un hecho determinado.

   La división entre estas categorías no es totalmente nítida, especialmente en los tres niveles superiores.

2. **Dominio afectivo.** Los objetivos pertenecientes a este dominio se refieren a actitudes, sentimientos, intereses y motivaciones. Se dividen en cinco categorías:
    - Recepción (atención): sensibilizar al alumno de la existencia de ciertos fenómenos.
    - Respuesta: participación activa del alumno, en su reacción ante algo.
    - Valorización: interiorización por parte del alumno de un conjunto de valores.
    - Organización: integración de diferentes valores hasta formar un sistema de valores con coherencia interna, mediante comparación, relación y síntesis de valores.

- Caracterización: influencia del sistema de valores sobre la conducta del individuo, de modo que se produzca una conducta característica de ese sistema de valores.

Los objetivos afectivos son más difíciles de evaluar que los cognitivos, al basarse en actitudes de las personas, que no son fáciles de determinar o de definir.

Así, aunque los objetivos se fijan principalmente en el plano cognoscitivo, es imprescindible que permitan a la vez fomentar actitudes como la motivación, la curiosidad, el espíritu crítico, la creatividad y el sentido práctico. La adquisición por parte de los alumnos de unos conocimientos que les permitan en el futuro desarrollar una actividad profesional ha de considerarse más como un medio que como un fin para la capacitación global del titulado.

Los conocimientos en sí no garantizan esta capacitación; el alumno aprende terminología y hechos demostrando estos conocimientos solamente recordando la información. Es preciso, por tanto, que el alumno asimile estos conocimientos, los comprenda y, más aún, sea capaz de obtener unos resultados a partir de los conocimientos adquiridos.

No hay que olvidar como objetivos del método de enseñanza la capacidad de análisis, síntesis y evaluación que ha de adquirir el alumno universitario. Entendiéndose como análisis la capacidad de observar la realidad y realizar una abstracción, obteniendo la relación con los conocimientos adquiridos. La síntesis permite el diseño de nuevos elementos por aplicación de los conocimientos adquiridos; es el máximo nivel que se puede fijar como objetivo en el ámbito de la carrera universitaria. La evaluación consiste en la deducción de una serie de características, comparándolas con unos criterios objetivos. En cuanto a la evaluación, el alumno puede indicar mejoras que se pueden hacer a los métodos o sugerir nuevas herramientas.

La formulación de los objetivos concretos facilita la elección de los contenidos del temario y del nivel al que se han de tratar. También permite una mejor programación de actividades y la elección del sistema de evaluación más apropiado.

Otro aspecto importante de la docencia es el de intentar que el alumno desarrolle de forma continua la asignatura. Los abandonos intermitentes en la atención a ésta disminuyen considerablemente su capacidad de obtener una visión global del conjunto, que es uno de nuestros principales objetivos. La consecuencia más perjudicial es la falta de asimilación de los conocimientos teóricos que permiten el desarrollo de las prácticas. Desde nuestro punto de vista, las prácticas no deben ser una materia aislada del resto de la asignatura, con contenidos autosuficientes, sino que deben ser utilizadas para afianzar los conocimientos adquiridos a través de las clases teóricas. Para alcanzar este objetivo es muy importante la motivación del

alumno, que sobre todo se puede conseguir con una adecuada planificación de las prácticas y con demostraciones prácticas que hagan que el alumno se interese por la asignatura.

2. Métodos docentes

El objetivo final de la enseñanza es el aprendizaje del estudiante. Para conseguir ese fin, el profesor utiliza métodos de enseñanza que le permiten organizar y desempeñar su función docente. Los métodos que se pueden utilizar dependen de la materia y del tipo del alumnado al que va destinada la enseñanza. Estos métodos utilizan una serie de técnicas que son formas de orientación inmediata del aprendizaje, procedimientos didácticos que facilitan el desarrollo específico de los métodos de enseñanza [5]. Los medios o recursos didácticos son los instrumentos que se utilizan para poner en práctica las técnicas.

La finalidad última del método docente debe contemplar no sólo el enriquecimiento de los conocimientos del alumno respecto al campo específico de la materia, sino que también debe potenciar el enriquecimiento personal para poder lograr una eficacia en un futuro profesional. Los métodos actuales de enseñanza se basan en tres principios de aceptación general:

- **Individualización.** Es la adecuación de las tareas educativas a la psicología del estudiante, considerando sus cualidades individuales y las diferencias existentes entre cada personalidad. Se tenderá a un tiempo de aprendizaje en el que cada alumno avanza en la adquisición de nuevos conocimientos según sus propias habilidades personales y su motivación personal. El profesor debe esforzarse por estimular a cada alumno, teniendo en cuenta sus intereses y progresos.

- **Actividad.** Es la necesidad de fundamentar el acto didáctico en la actividad del alumno. Durante mucho tiempo, los contenidos han jugado un papel predominante en el desarrollo de las clases; no es casualidad que la palabra lección tenga la misma raíz latina que la palabra lectura. Antes de la existencia de la imprenta, la lección consistía en la lectura o dictado de un manuscrito por parte del lector. La labor del alumno se limitaba a transcribir la información suministrada de forma que esta pudiera seguir transmitiéndose de generación en generación. Este panorama no es muy diferente al que se detecta cuando vemos a un alumno tomar apuntes en el aula sin apenas comprender lo que en ellos se dice, dejando esta tarea, en el mejor de los casos, para unas semanas antes del examen. No creemos que deba ser éste el método de enseñanza a seguir. El alumno debe ser un elemento totalmente activo en la vida académica. Esta actividad ha de estar encauzada, dirigida y estimulada por el profesor. El logro de dicha actividad se puede alcanzar con la utilización sistemática del diálogo en la clase,

mediante técnicas de trabajo en equipo y coordinando didácticamente las clases teóricas y prácticas que el alumno ha de realizar.

- **Participación.** La discusión de los objetivos y los métodos de enseñanza, con la aceptación de estos por parte de los alumnos, lleva consigo el logro de un clima mucho más estimulante, en el que el alumno desarrollará más profundamente sus capacidades, al dejar de estar sometido a presiones y autoritarismos innecesarios.

El método docente debe estar concebido teniendo en cuenta la necesidad de favorecer y estimular desde los centros de enseñanza la curiosidad, la observación, el razonamiento crítico, la capacidad de síntesis y la creatividad, sin olvidar una continua atención a los problemas técnicos que afectan a la sociedad, tanto para detectarlos como para plantearlos e intentar resolverlos.

La clasificación de los métodos de enseñanza puede hacerse utilizando distintos criterios. De entre todas las clasificaciones posibles, se incluyen en este proyecto aquellas que permiten clarificar e incluyen las distintas técnicas que se proponen para el desarrollo de la docencia de la asignatura objeto del presente proyecto.

1. Según la forma de razonamiento:
   Deductivo: se procede de lo general a lo particular.
   Inductivo: se parte de casos particulares y se intenta descubrir el principio general que los rige.
   Comparativo o analógico: los datos particulares que se utilizan permiten establecer analogías que llevan a una conclusión por semejanza (de lo particular a lo particular).

2. Según la coordinación de la materia:
   Lógico: los datos se presentan según un cierto orden predefinido.
   Psicológico: se sigue un orden no rígido, marcado por los intereses o experiencias del alumno.

3. Según la sistematización de la materia:
   Sistematizado: la clase sigue un guión.
   Ocasional: se aprovecha la motivación del momento y los acontecimientos importantes en la materia para orientar los temas de la clase.

4. Según la concreción de la enseñanza:
   Simbólico o verbalístico: todos los trabajos son ejecutados a través de la palabra.
   Intuitivo: se utilizan objetivaciones o concreciones para lo que se va tratando.

5. Según la actividad del estudiante:

Pasivo: el profesor imparte los conocimientos y los estudiantes permanecen en actitud pasiva.
Activo: el estudiante participa en el desarrollo de la clase bajo la guía del profesor.

6. Según la relación entre profesor y estudiante:
   Individual: el profesor enseña a un solo alumno.
   Recíproco: algunos alumnos enseñan a otros, actuando como monitores.
   Colectivo: el profesor enseña a muchos alumnos.

7. Según el trabajo del alumno:
   Individual: el trabajo realiza tareas diferenciadas de manera **individual**.
   Colectivo: el trabajo se realiza entre un grupo de alumnos.

La metodología tiene un carácter relativo y se fundamenta en razones lógicas y psicológicas; las primeras aportan criterios para estructurar el contenido y las segundas adecuan el contenido al contexto del currículum, a las características del alumno y a la psicología del aprendizaje.

En este proyecto se pretende una metodología activa, donde el alumno sea parte fundamental en su propio aprendizaje. Ello implica, por parte del profesor, una mayor capacidad para lograr una motivación orientada al trabajo, la creación de hábitos de estudio, investigación, trabajo en equipo, etc. Se tratará de fomentar la participación individual, en grupo y general, de forma que ninguna predomine excesivamente sobre las otras. Se tendrá también como objetivo propiciar en el alumno una actitud de búsqueda, descubrimiento y creación personal.

3. Técnicas de enseñanza

Los diferentes métodos de enseñanza toman en consideración una serie de aspectos que asignan un papel determinado al profesor, al alumno y a la materia a impartir, todo ello bajo distintos puntos de vista. Estos métodos o principios didácticos deben sustanciarse en las diversas técnicas de enseñanza que se emplean. Las técnicas de enseñanza se agrupan en tres categorías:

1. **Individuales.** Son las destinadas al alumno como individuo. Las más utilizadas son el estudio dirigido, las tutorías y las consultas bibliográficas. A veces, las tutorías pueden llevarse a cabo también en pequeños grupos, por lo que se pueden incluir, en estos casos, en la categoría siguiente.

2. **De grupo.** Los alumnos se organizan en pequeños grupos. Entre otras técnicas de grupo, las más relevantes para este proyecto docente son la técnica de proyectos, el diálogo simultáneo, los foros y los seminarios.

3. **De gran grupo.** Son las dirigidas a la clase en su conjunto. Las más utilizadas son la técnica expositiva y la pregunta.

A continuación, se describen brevemente algunas de las técnicas que se proponen en este proyecto y, con algo más de extensión, las que se consideran fundamentales para el desarrollo de la docencia.

Estudio dirigido. El alumno trabaja de forma individual sobre un tema según lo establecido por el profesor, que hace un seguimiento y orienta. El alumno aprende por sí mismo y profundiza más o menos en función de sus capacidades.

Técnica de proyectos. El profesor propone un tema y unos objetivos para la realización de un proyecto. Los grupos los pueden formar los alumnos o el profesor. Las tareas en cada grupo se realizan de forma individual o grupal. Se elabora un informe final del trabajo.

En el caso de utilizar esta técnica para la resolución de proyectos de desarrollo de software, ese informe final puede ser el sistema elaborado por el grupo. Esta técnica suele motivar al alumno, al enfrentarse a problemas reales. Sin embargo, tiene cierta dificultad en cuanto a la evaluación, ya que puede ser difícil determinar el grado en que ha trabajado cada alumno dentro del grupo.

Diálogo simultáneo. El profesor propone a los alumnos, en cualquier momento de la clase, que durante 2-3 minutos y de dos en dos, den solución a una cuestión o problema sencillo. Tras este tiempo, se presentan las distintas soluciones alcanzadas. Esta técnica sirve para permitir que los alumnos participen y se recupere su atención.

Seminario. En grupos de 5 a 12 alumnos, se trabaja durante varios días un tema, utilizando fuentes originales. Una vez determinado el tema los participantes buscan información, intercambian ideas y se elabora un documento. Suele haber un coordinador del grupo.

Tutorías

En un buen proceso de enseñanza y aprendizaje debe haber una estrecha relación entre el docente y el alumno. La estructuración actual de las enseñanzas universitarias contempla la posibilidad de que el alumno disponga de una atención individualizada del profesor en las llamadas horas de tutoría. Pueden utilizarse, entre otros fines, para resolver dudas de la asignatura, supervisar trabajos voluntarios realizados por los alumnos y supervisar problemas resueltos de manera individual.

Las tutorías son también de gran utilidad a los profesores, que tienen en ellas la oportunidad de realizar una autocrítica de la calidad de su docencia, especialmente si se detectan dudas generalizadas y defectos repetitivos en el aprendizaje de los alumnos.

Los horarios para las tutorías estarán fijados al principio de curso, de manera que se permita la asistencia del alumno. Serán de carácter no obligatorio.

Por otra parte, las tutorías pueden aportar, en algunos momentos, un beneficio para todos los estudiantes, ya que el profesor puede detectar en ellas la existencia de ciertas deficiencias conceptuales, pudiendo incidir y resolverlas en clases posteriores.

En general, son pocos los alumnos que hacen un buen uso de las tutorías a lo largo del curso y, en general, solo se suelen utilizar en las fechas cercanas a los exámenes. No obstante, esto está empezando a cambiar, gracias al mayor seguimiento e interacción que se está estableciendo en los últimos años por la utilización de herramientas online.

Técnica expositiva

La exposición del docente, es decir, la presentación y análisis de los contenidos que son objeto de aprendizaje es uno de los pilares del sistema metodológico del profesor. Esta técnica, también conocida como clase magistral, es el método por excelencia en la docencia universitaria.

Este era el método más habitual de enseñanza para grupos grandes antes de la llegada del Plan Bolonia, en el que el profesor trata de transmitir una serie de conocimientos a los alumnos utilizando fundamentalmente la palabra y la pizarra, el cañón o el retroproyector.

La actividad exigida a los alumnos es exclusivamente de atención, por lo cual es poco motivadora, capta difícilmente la atención del alumno, fomenta la pasividad en clase, y puede originar insatisfacción y aburrimiento. Los factores que más influyen en esta situación son:

- los alumnos reciben los mismos contenidos independientemente de su preparación y capacidad de asimilación.
- la masificación real.
- la interacción alumno-profesor se reduce al mínimo.
- no permite analizar los resultados hasta la fase de evaluación.

No obstante, la clase magistral es un buen método de introducción y está especialmente indicada para transmitir aspectos generales y conceptuales, y para síntesis de conocimientos. Una buena exposición magistral permite que el profesor transmita a los alumnos entusiasmo por la materia, creando en ella interés en el tema, incitándoles a la creatividad y llevándolos por el camino del rigor y la exactitud.

Conseguir las ventajas expuestas supone un esfuerzo considerable al profesor, que debe preparar con sumo cuidado cada una de las exposiciones, para mantener centrado el interés del alumnado. Para ello:

- Lo primero y esencial en la preparación de una clase magistral es determinar claramente cuáles son los objetivos, estableciendo qué se espera que el alumno sepa o sea capaz de hacer como resultado. Es importante no preparar más material del que pueda ser tratado cómodamente en el tiempo asignado.
- Desarrollar el tema de forma amena. En la presentación hay que procurar:
    - mantener el nivel de atención, usando ejemplos e intercalando puntualizaciones, en la medida que sea posible, para resaltar los conceptos fundamentales, así como alguna experiencia personal que pueda resultar interesante para el alumno; variando el ritmo y tono de voz para dar énfasis a lo más importante.
    - fomentar el diálogo profesor-alumno, aceptando preguntas u observaciones durante la explicación e interrogando al alumno por cuestiones que, aun no habiendo sido explicadas, puedan ser deducidas, fomentando el diálogo, el debate y la confrontación de ideas e hipótesis en clase.
    - apoyar las exposiciones de pizarra, con el uso de transparencias para la presentación de textos, esquemas, gráficos, etc., con lo que se evitan pérdidas de tiempo; presentando estos de forma ordenada y completa, lo que permite captar mejor la atención del alumno, centrarse sobre determinados aspectos y realizar comparaciones.
- En toda clase magistral se deberá poner de relieve la conexión de un tema con los que le precedieron y los que le han de seguir, para así dar al alumno una visión global de los temas que están relacionados y, si es posible, de toda la materia e incluso ir más allá, buscando relaciones entre distintas áreas y tipos de conocimientos.

La lección dialogada

En el intento de evitar las desventajas de la clase magistral, surge una variante denominada lección dialogada. En ella, al alumno se le exige una participación más activa y en la que se da paso a sus aportaciones personales sobre el tema. Con ello, es posible detectar el nivel de conocimiento del alumno sin tener que esperar a la evaluación final y controlar la asimilación de los conceptos expuestos. Incentiva la motivación de la clase y favorece el acercamiento afectivo del alumno.

La clase dialogada tiene su origen en la metodología didáctica basada en Aprendizaje Activo. Esta metodología hace referencia a un conjunto de aproximaciones a la educación que dan un mayor protagonismo al estudiante en el proceso de aprendizaje. El elemento común en estas aproximaciones es que el profesor suaviza su papel magistral de conductor frente a la clase,

induciendo a los estudiantes al autoaprendizaje o al aprendizaje por descubrimiento. La justificación fundamental de este tipo de propuesta se basa en que, aun cuando este no es el único elemento que interviene en el aprendizaje, el grado con que los alumnos comprenden un concepto está en proporción directa con el esfuerzo personal que han invertido en su aprendizaje. Una segunda justificación es que se asemeja más a la actividad que deberán emprender los estudiantes cuando se integren al mundo del trabajo. Además, esta técnica es especialmente adecuada para promover el aprendizaje en grupo, de forma que los estudiantes puedan organizarse para colaborar en la resolución de un problema común.

Para cumplir con esta metodología, en lugar de proporcionar directamente la solución a los ejercicios, se puede ir construyendo dicha solución a través de un conjunto de soluciones parcialmente incorrectas, que pueden o bien ser propuestas por los propios alumnos o ser facilitadas por el profesor. Los alumnos deben intentar descubrir los errores de estas soluciones y proponer nuevas alternativas que intenten eliminarlos.

La utilización de esta metodología se puede extender, en algunos casos, no sólo a la realización de ejercicios en el aula, sino a la presentación de los aspectos teóricos de la asignatura y a las prácticas de laboratorio.

La clase de problemas

Los ejemplos que pueden darse como ilustración a lo largo de las explicaciones teóricas son evidentemente insuficientes para desarrollar en el alumnado una idea adecuada de cómo aplicarlas. Por esta razón, es imprescindible dedicar clases al desarrollo de ejemplos concretos de aplicación en forma de ejercicios, problemas, cuestiones, etc. Durante la impartición de la clase magistral, en la cual se introducen nuevos conceptos, es difícil mantener una discusión razonable con los alumnos sobre cuestiones que acaban de ser explicadas.

Por las razones expuestas, se hace también necesaria la impartición de clases de problemas. En dichas clases, el alumno cuenta ya con una idea inicial del tema, de manera que se encuentra con posibilidades de hacer observaciones en el desarrollo de los ejercicios. Es en estas clases en las que la participación del alumno es más activa; la labor del profesor es actuar como director del diálogo que se pueda entablar, incitando a la participación.

Las clases de problemas han de ser un complemento esencial, orientadas a la realización de ejercicios que ayuden a la asimilación de estos conceptos, permitiendo al profesor corregir posibles deficiencias de comprensión. Es deseable que cada clase de problemas tenga unos objetivos preestablecidos. En función de estos objetivos se han de preparar los enunciados de los problemas a plantear, procurando que sean un reflejo de las consecuencias fundamentales de la teoría.

En la clase de problemas el alumno dispone de los ejercicios a realizar con anterioridad al desarrollo de la clase, y ha tenido tiempo de pensar y trabajar sobre ellos. De esta forma, hay un trabajo individual previo, sirviendo la clase de problemas para refrendar sus soluciones con las que explica el profesor o para aprender a resolver aquellos problemas que ha sido incapaz de resolver por sí solo.

Por supuesto, para que la clase de problemas sea realmente provechosa es necesario que los alumnos hayan trabajado sobre los ejercicios propuestos con anterioridad a la clase.

La mayor dificultad de las clases de problemas es la forma de impartir dichas clases. De hecho, no se debe enseñar a resolver problemas concretos, sino a estudiar soluciones de una forma general, que puedan servir al alumno como patrones para la resolución de problemas similares. Así, es necesario comenzar por un estudio cualitativo del tema, elaborando las estrategias de resolución, antes de proceder a la búsqueda de éstas, analizando distintas vías de resolución para permitir contrastar los resultados obtenidos y analizarlos a la luz de las hipótesis elaboradas. Es fundamental invitar a los alumnos a que aporten sus propias soluciones si es que son esencialmente distintas de las mostradas por el profesor.

En este caso, tal como se discute en [6], las soluciones deben ser analizadas dando los pros y contras de las mismas, de tal forma que el alumno comprenda que para la resolución de un problema no hay sólo un camino y lo realmente importante no es el problema en sí, sino las herramientas y técnicas que conducen a su solución.

Las clases de laboratorio

Las prácticas con el ordenador constituyen un elemento fundamental en la enseñanza de las ciencias de la computación. Según el profesor Knuth [7], no puede decirse que se ha aprendido bien una materia hasta que se es capaz de enseñarla a un ordenador, es decir, expresarla mediante un algoritmo.

Las clases de laboratorio son de gran importancia en la formación de futuros Ingenieros del Software, ya que constituyen el complemento necesario de las clases de teoría y problemas. Las prácticas son también un medio excelente para que el alumno potencie su iniciativa y capacidad crítica. Además, son imprescindibles porque proyectan conocimientos acumulados sobre problemas reales. En el laboratorio se establece un ambiente más próximo al entorno profesional. En general, las prácticas en laboratorio han de permitir comprobar la capacidad real del alumno para llevar a la práctica sus conocimientos teóricos y de forma creativa. Además, la gran diversidad de problemas a los que ha de enfrentarse el alumno, muchos de ellos provenientes de resultados empíricos, y la finalidad puramente práctica con la que ha de hacerlo, hacen que el laboratorio sea algo más que la mera verificación práctica de los

resultados de la teoría. En muchas ocasiones, son las clases de laboratorio las que impulsan al alumno a investigar en la teoría, y las que deben inculcarle el espíritu crítico que debe estar presente en cualquier ingeniero.

Por todo ello, es fundamental preparar las clases de laboratorio con una fuerte dedicación y cuidado, de manera que se propongan las prácticas que resulten más útiles a los alumnos.

En las sesiones de laboratorio, en función del número de alumnos, se podrán realizar prácticas individuales o en grupo. En estas últimas, se formarán grupos de dos a cuatro alumnos. Antes de que los alumnos comiencen, se les proporcionará un programa completo de información sobre el desarrollo de estas, de manera que contenga los objetivos generales y el material de trabajo.

Es importante la sincronización que debe existir entre las clases de teoría y las clases de laboratorio, de forma que el alumno conozca los aspectos teóricos previamente al planteamiento práctico. Al igual que los ejercicios de las clases de problemas, las prácticas deben formularse con suficiente antelación, para que los alumnos puedan prepararlas. Los alumnos deberán elaborar sus propias soluciones y desarrollar el trabajo práctico en el laboratorio.

En general, cualquier asignatura de laboratorio en una titulación de ingeniería debe estar orientada a que los alumnos:

- Construyan prototipos de los modelos explicados en teoría.
- Comprueben de forma empírica los problemas que se plantean en la materia explicada, justificando así las soluciones planteadas y comprendiendo las razones que llevaron a descartar otras.
- Experimenten nuevas soluciones y analicen los resultados obtenidos.
- Desarrollen capacidades analíticas que les permitan buscar las razones que justifican los resultados obtenidos, sobre todo cuando éstos no concuerdan con los que podrían esperarse.
- Realicen trabajos en grupo, donde el éxito del proyecto dependa de todos y cada uno de los participantes, de forma que aprendan la importancia del trabajo en equipo.

En cuanto a los conocimientos que se pretende inculcar al alumno en la asignatura objeto de este proyecto, las clases de laboratorio persiguen, en particular, dos grandes objetivos:

- Aplicar y reforzar los conocimientos adquiridos en las clases teóricas.
- Desarrollar y consolidar la disciplina de programación.

Consultas bibliográficas

La consulta de bibliografía es un aspecto didáctico fundamental en la enseñanza universitaria. La selección de la bibliografía es una parte de la labor del profesor tan importante como la elaboración de un temario. Dirigir al alumno hacia grupos escogidos de consulta y estudio incentiva la actitud investigadora del mismo, a la vez que desarrolla en él una actitud crítica.

La selección de bibliografía propuesta para los distintos temas (en el capítulo de programación docente) se ha realizado teniendo en cuenta los siguientes criterios:

- Ser asequible: el alumno debe poder disponer de las referencias que se dan. Inconvenientes relacionados pueden ser los medios de distribución, los horarios de biblioteca, etc.
- No debe ser excesivamente abundante, para que el alumno no se desanime. Es conveniente, además, jerarquizarla en un orden natural de consulta.
- Debe haber una diversidad de niveles de dificultad, de manera que haya unos textos que se adapten al nivel de los temas tratados en clase y otros que permitan su ampliación.

4. Medios para la docencia

Los medios más utilizados en la docencia son, entre otros: pizarra, documentos escritos, sistemas de presentación con ordenador, vídeos, demostraciones con ordenador y el Campus Virtual, que incluye una serie de herramientas digitales cuyo uso es cada vez más amplio. De entre ellos, se describen aquellos que tiene mayor interés en este proyecto docente por su adecuación al mismo y su amplia disponibilidad.

Sistemas de presentación con ordenador

Son sistemas informáticos que permiten diseñar y utilizar apoyos visuales en la exposición. Permiten organizar la información y transmitirla de forma atractiva y comprensible. Se utilizan en conjunción con un medio de proyección. Además, las presentaciones que se van a utilizar en las clases pueden estar disponibles con antelación para los alumnos, de cara a facilitar el seguimiento de estas.

Demostración basada en ordenador

La disponibilidad de ordenador y proyector en el aula permite un uso más general que otros sistemas de presentación (como la pizarra) ya que puede utilizarse para mostrar cualquier tipo de información para ilustrar los contenidos de la asignatura. En el caso de la asignatura objeto de este proyecto docente, la posibilidad de escribir, compilar y ejecutar programas en el aula

es una importante motivación para los estudiantes, ya que les permite comprender mejor el proceso.

El campus virtual

En la Universidad de Málaga, desde hace ya varios años se dispone de un campus virtual, basado en la plataforma Moodle. El campus ofrece un espacio virtual para cada asignatura, donde es posible compartir ficheros, asignar actividades, discutir usando un foro, etc. El campus se usa muy activamente en la asignatura de este proyecto docente y se ha convertido en parte integral del proceso.

5. La evaluación

El proceso de enseñanza y aprendizaje tiene por objeto producir ciertos cambios en el alumno, consiguiendo así unos objetivos determinados. La evaluación consiste en comprobar si se han conseguido estos objetivos con dicho proceso. En la evaluación se pueden diferenciar dos etapas: determinar el conjunto de conocimientos y aptitudes de cada alumno y decidir si estos son o no suficientes. Las primeras tareas son juicios de hecho, de carácter objetivo; las últimas, en cambio, pertenecen al ámbito de lo opinable.

La evaluación supone la emisión de un juicio de valor acerca de algo o de alguien. En la enseñanza, la evaluación puede referirse al alumno (su aprendizaje) o al profesor (su enseñanza).

La evaluación tiene tres componentes:

- Información, que debe ser adecuada.
- Juicios, basados en la interpretación de la información anterior.
- Decisiones, que se fundamentan en los mencionados juicios.
- La información es un conjunto de datos respecto a aquello que se va a evaluar. Estos datos deben ser válidos y fiables. Válidos porque deben estar en correspondencia con lo que se pretende evaluar, de modo que sean apropiados para los juicios de valor que se deben emitir y las decisiones que se van a tomar. Fiables porque deben estar exentos de error o el error debe ser el mínimo posible.

La evaluación se aplica a distintas fases del proceso educativo universitario. Deben evaluarse los siguientes aspectos:

- La situación del alumno al acceder a una asignatura.
- La medida en que los alumnos alcanzan una formación adecuada a las expectativas u objetivos de la carrera o de la asignatura.

- La adecuación y calidad de las previsiones de la actividad formativa, es decir, su programación.
- La acción en el aula y la comunicación entre el profesor y los alumnos.
- La evaluación en sí misma también debe ser sometida a análisis y valoración.

Para justificar la existencia de la evaluación, será preciso estudiar sus consecuencias favorables y desfavorables, y ver cuáles son las que predominan. Se pueden estudiar los efectos de la evaluación bajo tres perspectivas distintas: desde el punto de vista del alumno, del profesor y de la sociedad.

Sobre el alumno, los efectos que pueden producirse son los siguientes:

- Toma de conciencia del propio conocimiento, de su amplitud, sus límites y sus lagunas. Sólo la evaluación pone ante los ojos del alumno qué, cómo y cuánto sabe.
- Satisfacción por la evaluación positiva, lo que constituye un estímulo para tener en lo sucesivo la misma conducta.
- Insatisfacción por una evaluación negativa o inferior a la esperada, lo cual puede ser un estímulo para variar de conducta.
- Temor de las consecuencias de una evaluación totalmente negativa, que puede llevar al rechazo y abandono de la asignatura, con la lógica perturbación del proceso de aprendizaje.
- Distorsión de la actividad docente cuando el objetivo prioritario del alumno se centra en obtener una evaluación positiva y no en el aprendizaje de la materia.

En cuanto al profesor, los efectos que pueden generarse son los siguientes:

- Conocimiento del estado medio del alumnado en lo que a la enseñanza se refiere. Este conocimiento sirve para adaptar la actividad docente global a las necesidades detectadas y para reconsiderar la programación y los métodos, por si hubiera que modificarlos en cursos venideros. Para ello, es conveniente la elaboración de estadísticas y encuestas que pulsen la opinión del alumno sobre la propia evaluación.
- Conocimiento del estado de aprendizaje de cada alumno en particular.

En cuanto a la sociedad en su conjunto, los efectos de la evaluación son:

- Garantizar que todo alumno que sale de la Universidad posee un mínimo de conocimientos, capacidad de estudio y potencialidad de afrontar nuevos problemas.
- Informar sobre el grado de dichos conocimientos y aptitudes.

El aprendizaje universitario no se produce de forma espontánea, es necesario un esfuerzo por parte del alumno. Por tanto, es necesario proporcionarle un estímulo para que realice dicho

esfuerzo. Hoy por hoy, el mayor estímulo lo constituye la evaluación. Sería mejor, nadie lo discute, que los estímulos predominantes fueran otros: curiosidad intelectual, formación para el ejercicio de la profesión, etc.; pero no parece realista, en nuestras circunstancias concretas, prescindir del refuerzo que las técnicas de evaluación proporcionan.

De hecho, lo que los alumnos suelen cuestionar no es la existencia de evaluaciones, sino los resultados inexactos que en ellas se pueden presentar. En efecto, de las dos tareas antes señaladas, la determinación del estado de aprendizaje del alumno es una cuestión bastante compleja. Además, se debe tener en cuenta que la insatisfacción causada por una evaluación negativa puede ser interpretada psicológicamente como injusta por el alumno que cree que sabe, cuando en realidad la evaluación ha puesto de manifiesto su falta de conocimiento en determinadas cuestiones.

En cuanto al juicio de valor implicado en el proceso de calificación de un alumno, es decir, en la aceptación o el rechazo del grado de aprendizaje detectado en el mismo, la cuestión es aún más difícil. Se puede obtener una demostración concluyente de la ignorancia de un punto concreto, por parte de un alumno; no así de lo inadmisible de esta ignorancia.

Para paliar estos efectos, la evaluación ha de ser lo más extensa posible en cuanto a la materia exigida y los criterios de valoración han de estar establecidos y deben ser conocidos por los alumnos previamente.

Una razón más en favor de la existencia de la evaluación la proporciona la sociedad misma. La sociedad recibe de la Universidad cierta información relativa a los alumnos que han pasado por sus aulas, constituida por el conjunto de materias cursadas y las calificaciones en ellas obtenidas.

La evaluación de los conocimientos de la materia adquiridos por el alumno es, por lo general, una tarea difícil, más aún cuando se trata de grupos numerosos. Entre las técnicas de evaluación que normalmente se emplean, en este proyecto se propone la utilización de las dos siguientes:

- Exámenes. Son quizás la forma más característica de evaluación en las universidades españolas, entendiéndose estos como pruebas de evaluación que se realizan en condiciones controladas de tiempo, lugar y comunicación. Es sabido por todo aquel que ha realizado exámenes que la nota obtenida no siempre refleja el grado de conocimientos que de la materia se tiene. Los factores de azar, la ansiedad y el nerviosismo que provocan en ciertos alumnos, así como la picaresca estudiantil, tergiversan en muchos casos los resultados. Sin embargo, los exámenes tienen como ventaja innegable la objetividad originada por las formalidades que los rigen y la facilidad de valoración por parte del evaluador. Es, sin duda, por estas razones por las

que este método se perpetua en nuestras universidades. Según la actividad que se exige del alumno, se pueden señalar los siguientes tipos de pruebas o ejercicios:

- Exposición de lecciones, temas o epígrafes del programa original de la asignatura.
- Respuestas breves, en formato libre, a cuestiones concretas referentes a los conceptos y técnicas fundamentales de la materia.
- Selección de una o varias de entre las respuestas que se ofrecen a una cuestión concreta y sin ambigüedades. Estos últimos ejercicios suelen denominarse tipo test.
- Desarrollo de soluciones a situaciones problemáticas.

Cada uno de estos ejercicios tiene sus ventajas y sus inconvenientes. Con los primeros se puede evaluar el conocimiento adquirido por el alumno y la comprensión de la materia, pero no la asimilación de los conceptos para elaborar a partir de ellos soluciones a situaciones nuevas. Con ciertos tipos de ejercicios se puede abarcar un espectro más amplio de temas, mientras que las preguntas cortas o de test resaltan los conocimientos más específicos. Por último, un punto a considerar es la influencia que el tipo de examen tiene sobre el método de estudio del alumno. Un examen muy teórico se presta más a un estudio basado en la pura memoria, mientras que con los ejercicios de problemas se fomenta la realización de prácticas que permiten una mejor comprensión de la materia. En la práctica, es conveniente la alternancia de distintas modalidades de ejercicios para motivar al alumno hacia un estudio completo de la materia, aunque, por lo general, deberán predominar los ejercicios de resolución de problemas, ya que son más completos y, por tanto, se adecuan mejor a la evaluación de los objetivos globales de la asignatura.

- Valoración de trabajos relacionados con la materia realizados por el alumno, de manera individual o en grupo. Esta técnica tiene como ventaja sobre los exámenes la continuidad y la menor presión psicológica inducida en el alumno. La realización de trabajos en grupos permite que la complejidad de estos sea mayor y sirve como entrenamiento de los alumnos en uno de los aspectos más importantes hoy en día en los proyectos de desarrollo en informática: el trabajo en equipo. Además, el grupo puede servir como elemento de interrelación e intercambio de conocimientos. Por otra parte, la realización de trabajos individualmente tiene como ventaja indiscutible la mayor justicia en la nota obtenida en los mismos por un individuo concreto, ya que la nota obtenida depende exclusivamente de él. La realización de trabajos individuales como método de evaluación es muy difícil en la práctica dado el gran número de estudiantes que hay en la actualidad. Incluso la evaluación de trabajos en grupos es problemática.

Es, por tanto, difícil realizar una evaluación completa mediante esta técnica. Sin embargo, la realización de trabajos puede complementar y refinar la evaluación que se obtiene mediante el método de exámenes. En la práctica, la realización de los trabajos es voluntaria y no excluye de la necesidad de superar el examen, pero sirve para que aquellos alumnos que demuestran interés especial por la asignatura obtengan en ella mejores calificaciones.

PARTE III. PROGRAMACIÓN DOCENTE

En este capítulo se detalla la programación docente de la *asignatura Introducción a la Ingeniería del Software*, del segundo curso de la titulación de Graduado/a en Ingeniería del Software. Dicha asignatura es de 6 créditos ECTS y tiene carácter obligatorio dentro del plan de estudios de la titulación.

La elaboración de esta programación docente tiene como base la programación actual de la asignatura, coordinada por el profesor Antonio Maña Gómez. También ha influido la experiencia del candidato en asignaturas con contenidos similares impartidas en otra universidad. Particularmente, ha impartido durante tres cursos académicos la asignatura *Software Engineering Project*, obligatoria de 20 créditos en el segundo curso del *BSc. In Computer Science* de la Universidad de York, en el Reino Unido, donde anualmente se matriculan unos 150 alumnos.

1. La Ingeniería del Software

Antes de entrar a desarrollar la programación docente de la asignatura, es necesario tratar de establecer una definición y definir el ámbito de la Ingeniería del Software como disciplina.

El término ingeniería del software apareció por primera vez en la década de 1950 y principios de los años 1960. El uso de este término se extendió durante las décadas de 1960-80, en las que se produjo la llamada *crisis del software*, que identifica muchos de los problemas de desarrollo de software a gran escala. Muchos de estos proyectos de software sobrepasaron el presupuesto y el tiempo inicialmente estimados para su desarrollo. Algunos proyectos causaron daños a la propiedad e incluso en algunos casos, la pérdida de vidas humanas. Aunque la crisis del software fue originalmente definida en términos de productividad, evolucionó hasta abarcar también aspectos relacionados con la calidad del software.

Como respuesta a esta crisis, la ingeniería del software surgió como una disciplina que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo. A lo largo de estas últimas décadas, se ha comenzado a considerar al ingeniero software como una profesión implantada en el mundo empresarial y con un claro reconocimiento social.

La ingeniería del software trata con áreas muy diversas de la informática y de las ciencias de la computación, tales como la construcción de compiladores, sistemas operativos, o desarrollo de sistemas distribuidos, abordando todas las fases del ciclo de vida del desarrollo de cualquier tipo de sistemas de información y aplicable a muchos ámbitos como el de los negocios, la medicina, investigación científica, banca, etc.

Hoy en día, la ingeniería del software se ha convertido en una disciplina que cada vez abarca más ámbitos debido a la ubicuidad del software y a su capacidad como fuerza transformadora de la realidad en la que vivimos. Desde los vehículos que conducimos hasta la gestión de nuestros impuestos, prácticamente todos los sistemas tecnológicos de los que dependemos contienen una cantidad sustancial de software. Esta tendencia no hace más que crecer, y nuevos desarrollos que vemos aparecer en nuestro horizonte vital, desde los vehículos autónomos y sistemas robóticos, hasta ciudades inteligentes, incorporarán cada vez más y más software como piedra angular de su funcionamiento. Es por ello que es fundamental seguir desarrollando esta disciplina para afrontar los retos actuales y futuros, así como formar a las nuevas generaciones de ingenieros software para que tengan una visión amplia de la disciplina y un conocimiento profundo de las problemáticas que afectan al software que cimente el desarrollo de sus futuras carreras profesionales y avance el estado de la práctica.

Una vez definido el concepto y ámbito de la Ingeniería del Software, pasamos a desarrollar el proyecto docente de da asignatura en el contexto del plan de estudios del Grado de Ingeniería del Software de la Universidad de Málaga.

2. Descripción de la asignatura

Grado:	*Graduado/a en Ingeniería del Software*
Centro:	*Escuela Técnica Superior de Ingeniería Informática*
Universidad:	*Universidad de Málaga*
Asignatura:	*Introducción a la Ingeniería del Software*
Código:	*206*
Departamento:	*Lenguajes y Ciencias de la Computación*
Área:	*Lenguajes y Sistemas Informáticos*
Tipo:	*Obligatoria*
Materia:	*Ingeniería del Software, Sistemas de Información y Sistemas Inteligentes*
Módulo:	*Formación común*
Experimentalidad:	*69 % teórica y 31 % práctica*
Idioma:	*Inglés, Español*
Curso:	*2*
Semestre:	*2*

Nº Créditos ECTS:	*6*
Horas de dedicación del estudiante:	*150*
Tamaño del Grupo Grande:	*72*
Tamaño del Grupo Reducido:	*30*

3. Competencias

De las competencias generales, básicas y especificas recogidas en el plan de estudios del Grado en Ingeniería del Software, se desarrollan en esta asignatura las siguientes:

3.1. Competencias generales y básicas

3.1.1 Competencias básicas

CB02. Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.

CB05. Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

3.1.1 Competencias generales

CG01. Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.

CG05 Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en las competencias básicas, comunes y específicas del título.

CG08 Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

CG09 Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero Técnico en Informática

3.2. Competencias específicas

CC01. Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.

CC16. Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software.

4. Contenidos

Los contenidos de la asignatura se organizan en los siguientes siete módulos

1. Introducción a la Ingeniería del Software - *1 sesión*

1.1. Justificación de la Ingeniería del Software

1.2 Definiciones

1.3 Características del software

1.4 Mitos del software

1.5 Aspectos éticos del software

En este módulo de introducción se presentan los conceptos básicos de la asignatura, justificando la necesidad de una ingeniería del software, a la vez que se introduce la evolución histórica en los procesos de creación de software a lo largo del tiempo. Se presenta la crisis del software, así como el concepto de ciclo de vida o modelo de proceso del software y algunos ejemplos de estos (p. ej. cascada, espiral). Se dedica una parte de este módulo a desmontar algunos de los mitos que existen sobre el desarrollo de software. También se presenta una categorización general del software y se tocan algunos de los aspectos éticos relacionados con el desarrollo de software.

La duración estimada de este módulo es de una sesión de dos horas de duración, que se desarrolla habitualmente en el aula. El formato elegido para la docencia es el de lección dialogada, en la que se fomenta continuamente la participación del alumno para que aporte sus conocimientos previos y sus puntos de vista, de manera que sirva para una primera toma de contacto entre el alumno y el profesor, fomentando la interacción entre ambos y

permitiendo a este último hacerse una idea de cuáles son los conocimientos tanto teóricos como prácticos de los alumnos en el ámbito de la asignatura.

2. Procesos software - *2 sesiones*

2.1. Actividades dentro de un proceso software

2.2 Modelos de proceso software

2.3 Modelos clásicos

2.4 Modelos especializados

2.5 Métodos ágiles

En este módulo se presentan la definición y los conceptos básicos relacionados con los procesos software, describiendo los distintos tipos que existen, la finalidad de cada uno de ellos, y la relación que existe entre los mismos de acuerdo con diferentes autores como Pressman y Sommerville. En cuanto a modelos de procesos software o de ciclo de vida, se hace un repaso a algunos modelos clásicos (cascada, espiral, RUP), especializados (V-Model, basado en compoenentes), y también a los métodos ágiles como Scrum y XP.

La duración estimada de este módulo es de dos sesiones de dos horas de duración, que se desarrollan en el aula. El formato elegido para la docencia es de nuevo el de lección dialogada, en la que se fomenta continuamente la participación del alumno para que aporte sus conocimientos previos y sus puntos de vista sobre las cuestiones que se plantean.

3. Gestión de proyectos software - *3 sesiones*

3.1. Gestión de proyectos software

3.2. Actividades de la gestión

3.3. Planificación de proyectos

3.4. Programación temporal de proyectos

3.5. Gestión del riesgo

3.6. Control de versiones - GiT

Este módulo comienza por describir los factores principales que influyen en el éxito de un proyecto software, así como los aspectos económicos de la ingeniería del software y las características que hacen única a la gestión de los proyectos software, comparada con la gestión de proyectos en otras disciplinas. A continuación, se da un repaso a las diversas actividades de gestión involucradas y se da un repaso general al proceso de planificación de proyectos, introduciendo el concepto de plan de proyecto, su estructura, y conceptos importantes relacionados como los hitos (*milestones*) y entregables (*deliverables*). Se introduce la programación temporal de proyectos y se mencionan algunos de los problemas

típicos que suelen surgir durante la misma. Se introducen herramientas básicas como los diagramas de Gantt y Pert, y conceptos asociados relevantes como el de camino crítico o *critical path*, con ejemplos. Por último, se dan algunas nociones básicas sobre gestión del riesgo, desglosando las distintas fases del proceso e identificando los distintos tipos de riesgo que se pueden dar (tecnológicos, de personal, etc.). Al final del tema se hará un repaso al control de versiones y a la herramienta GiT.

La duración de este módulo es de tres sesiones. Dos de ellas transcurren en el aula de teoría en formato de lección dialogada, y la otra en el aula de prácticas. En este módulo comienzan las prácticas de la asignatura, que siguen un enfoque basado en proyectos y que los alumnos realizan en grupos de 6 a 8 integrantes. La práctica correspondiente a este módulo consiste en adquirir las nociones prácticas necesarias para realizar la planificación preliminar del proyecto que se tendrá que realizar a lo largo de la asignatura. Para esto, se realizará una práctica en manejo de la herramienta Trello.

4. Ingeniería de requisitos - *3 sesiones*

4.1. La ingeniería de requisitos

4.2. Requisitos funcionales y no funcionales

4.3. Especificación de requisitos

4.4. Proceso de ingeniería de requisitos

4.5 Obtención y análisis de requisitos

4.6 Validación de requisitos

Tras ver algunas nociones generales tales como los requisitos, sus funciones, y tipos de acuerdo con su nivel de abstracción (usuario vs. sistema), se pasa a hacer una introducción a las nociones de requisito funcional y no funcional, dando ejemplos y algunas indicaciones que serán necesarias para la realización de las prácticas. A continuación, se describe el proceso de especificación de requisitos, así como los distintos documentos que suelen recoger los requisitos, y su relación con el diseño. Se presentan diversos métodos para la descripción de requisitos (lenguaje natural, plantillas), así como sus ventajas e inconvenientes.

Tras la especificación, se presenta el proceso de ingeniería de requisitos con sus diversas fases de estudio de viabilidad, elicitación, análisis, validación, y gestión, entrando en detalles sobre cómo se llevan a cabo cada una de estas fases. Se hace especial énfasis en la obtención y análisis de requisitos, describiendo algunas de las técnicas más utilizadas con más frecuencia, así como algunos de los problemas que pueden presentarse comúnmente.

La duración de este módulo es de tres sesiones. Dos de ellas transcurren en el aula de teoría en formato de lección dialogada, y la otra en el aula de prácticas. La práctica correspondiente a este módulo consiste en realizar una entrevista con el cliente (profesor de la asignatura), del que tendrán que extraer la información necesaria para la elicitación del requisitos del proyecto a desarrollar durante la asignatura.

5. Modelado con UML - *6 sesiones*

5.1. Modelado de sistemas

5.2. Casos de uso

5.3. Diagramas de clase

5.4. Diagramas de secuencia

En este módulo se comienza introduciendo el proceso de modelado y su contexto, presentando el concepto de modelo, así como los criterios necesarios para establecer su validez, y su propósito. Se hace un repaso por las distintas perspectivas de un sistema (externa, de interacción, de comportamiento, estructural) y se hace una introducción al lenguaje UML como ejemplo de lenguaje que hace uso de notaciones gráficas para la representación de las distintas perspectivas del software. Concretamente, se presentan las notaciones para los modelos de contexto, de interacción, estructurales, y de comportamiento.

A continuación, se presentan los diagramas de clase, introduciendo todos los conceptos relevantes relacionados como clases y objetos, relaciones, agregación, composición, y generalización. Por último, se presentan los diagramas de secuencia, presentando algunos patrones importantes con ejemplos (p.ej. comunicación síncrona, asíncrona, mensajes encontrados, bucles, paralelos).

El módulo está compuesto de 3 sesiones de teoría, y 3 de prácticas (todas de 2 horas de duración). Las sesiones de teoría discurrirán en el aula mediante lección dialogada. Para los contenidos prácticos, habrá una primera sesión en la que se trabajarán los casos de uso, mientras que en la segunda sesión será el turno de los diagramas de clases. Por último, durante la tercera sesión se tratarán los diagramas de secuencia.

6. Diseño software - *4 sesiones*

6.1. Arquitecturas software

6.2. Diseño software

En este módulo se comienza presentando el concepto de arquitectura software, comenzando varias definiciones alternativas y argumentando su importancia en el contexto del desarrollo

de sistemas software complejos. Se presentan los principales conceptos relacionados como la noción de componente, conector, o configuración, y se introduce la integridad conceptual y el concepto de patrón arquitectónico. A continuación, se ven algunos de los estilos clásicos de arquitectura como basados en flujos de datos (p.ej, *pipe and filter),* basados en llamada-retorno, o por capas, entre otros.

También en esta unidad se ve el concepto de diseño software, así como sus principios básicos. Como caso particular y relevante, se introducen los principios básicos del diseño orientado a objetos, así como el concepto y un catálogo básico de patrones de diseño.

Este módulo contiene 3 sesiones de teoría y 1 sesión práctica. Mientras que las sesiones de teoría serán en formato lección dialogada en el aula teórica, la sesión práctica se dedicará a patrones de diseño.

7. Verificación y pruebas - *3 sesiones*

7.1. Tipos de pruebas

7.2. Pruebas de caja blanca

7.3. Pruebas de caja negra

7.4. Test-Driven Development (TDD)

7.5. Pruebas unitarias / JUnit

7.6. Mocking / Mockito

En este módulo se presentan la verificación y pruebas de software, comenzando con un repaso histórico a los errores de software y algunos casos famosos. A continuación, se presentan algunos conceptos básicos como el de error y caso de pruebas, estableciéndose el contexto general de las pruebas de software (*software testing*), así como su finalidad. Después, se presentan los distintos tipos de prueba, de acuerdo con su finalidad (unitarias, de integración, regresión, etc.), y se describen brevemente algunas de las herramientas disponibles para realizarlas. Se presentan también las pruebas de caja blanca, así como los conceptos de cobertura de código, complejidad ciclomática, y camino base. Después, se presentan las pruebas de caja negra y se hace una breve presentación del diseño dirigido por pruebas (*Test-driven Development* o TDD). Por último, se introducen las pruebas unitarias con *JUnit* y el *mocking* con *Mockito*, respectivamente.

En este último módulo hay 2 sesiones de teoría, y una sesión práctica, todas de dos horas de duración. Mientras que las sesiones de teoría serán en formato lección dialogada, la sesión práctica tratará sobre el desarrollo de capacidades en prácticas unitarias practicando con

JUnit, y del mo*cking* con *mockito*, aplicando los conocimientos adquiridos durante la segunda sesión teórica de este bloque.

5. Bibliografía y otros recursos recomendados

- El Lenguaje Unificado de Modelado; Booch G., Rumbaugh J., Jacobson I.; Addison-Wesley; 2006
- El proceso unificado de desarrollo de software; Jacobson, I., Booch G., Rumbaugh, J.; Addison Wesley; 2000
- Ingeniería de Software Orientada a Objetos con UML, Java e Internet; Weitzenfeld A.; Thomson; 2005
- Ingeniería de Software; Sommerville, I.; Addison-Wesley Iberoamericana; 2005
- Ingeniería del Software. Un Enfoque Práctico; Pressman R. S.; McGraw-Hill; 2010
- DUM: Desarrollo Unificado con Métrica; Peláez, J.I., Gámez, J.I., Doña, J.; Universidad de Málaga; 2008
- Scrum y XP from the Trenches. Kniberg, H. InfoQ; 2007

6. Actividades Formativas

Las actividades presenciales de la asignatura se dividen en sesiones de grupo grande (GG), correspondientes a la tradicional lección magistral y sesiones de prácticas en grupo reducido (GR). Corresponden a las sesiones de grupo grande unas 40 horas, mientras que a las de grupo reducido unas 20 horas. En estas horas se cuentan las pruebas teóricas llevadas a clase en horario de clase, así como las evaluaciones prácticas.

No obstante, dado el carácter eminentemente práctico de los contenidos de la asignatura, y exceptuando el primer módulo, tanto unas como otras se realizan en el aula de informática, de manera que el alumno puede ir practicando los conceptos, técnicas y herramientas a medida que son presentados por el profesor.

Con el número de alumnos que cursan esta asignatura los últimos años, se determina que exista un único grupo grande y dos grupos reducidos. Semanalmente se imparte una sesión de GG y dos sesiones de GR, a cada una de las cuales acude la mitad de los alumnos. Las sesiones de clase son de dos horas. Todo ello, junto con la división del semestre en 15 semanas, determina que se deban realizar algunos ajustes en los que se favorece siempre la docencia en grupo reducido, pero en ningún caso superando las 60 horas totales presenciales del alumno, ni las cuatro horas por semana.

Las sesiones de grupo reducido se dedican fundamentalmente a la realización de prácticas en grupos de 3 o 4 alumnos. El enfoque seguido es el desarrollo de un proyecto siguiendo el denominado método del caso [3]: los contenidos son explicados y ejercitados sobre un proyecto real de desarrollo de software, que se presenta a los alumnos a principio de curso, a cargo de un profesional externo a la universidad y que ejerce la figura de cliente (o por el profesor, en caso de que no sea posible encontrar un profesional externo disponible). Además, se considera fundamental formar a los alumnos en la competencia de trabajo en grupo, dado que la inmensa mayoría de los desarrollos profesionales de Ingeniería del Software se realizan en equipo. Por este motivo los alumnos han de desarrollar en grupo el proyecto anteriormente citado.

De acuerdo con esto, al comienzo del curso se plantea a los alumnos un supuesto práctico que estos van desarrollando a lo largo del curso, de acuerdo con los contenidos y tecnologías correspondientes a cada uno de los módulos.

Por su parte, las actividades formativas no presenciales suman 85 horas, de las cuales se estima que 25 horas son de trabajo individual del alumno (estudio, instalación y aprendizaje del uso de las distintas herramientas), y los 60 restantes se dedican al desarrollo de las prácticas en grupo.

A estas horas se suman 5 horas de evaluación final, resultando un total de 150 horas estimadas de dedicación del alumno, tal como corresponde a una asignatura de 6 créditos ECTS.

7. Cronograma de la asignatura

El cronograma de la asignatura varía ligeramente de año en año en función del calendario académico y los días festivos que existan en el semestre de docencia. A modo de ejemplo, se incluye un cronograma de 15 semanas, que suelen ser las semanas completas de las que se dispone. Cada semana hay dos sesiones de dos horas cada una (no se considera en esta planificación el desdoble de grupos), obteniendo un total de 60 horas presenciales.

Semana	Sesión
1	Presentación de la asignatura
	Tema 1 – Introducción
2	Tema 2 – Procesos software (1/2)
	Tema 2 – Procesos software (2/2)

3	Tema 3 – Gestión de proyectos
	Tema 3 – GiT
4	Tema 4 – Requisitos (1/2)
	Práctica de Tema 3 – Control de versiones / GiT
5	Tema 4 – Requisitos (2/2)
	Práctica de Tema 3 – Planificación / Trello
6	Práctica de Tema 4 – Requisitos
	Tema 5 – Introducción a UML y casos de uso
7	Tema 5 – Diagramas de clases
	Práctica de Tema 5 – casos de uso
8	Tema 5 – Diagramas de secuencia
	Práctica de Tema 5 – diagramas de clases
9	Tema 6 – Arquitectura software
	Tema 6 – Diseño software (1/2)
10	Tema 6 – Diseño software (2/2)
	Práctica de Tema 5 – diagramas de secuencia
11	Tema 7 – Introducción a pruebas de software
	Práctica de Tema 6 – patrones de diseño
12	Tema 7 – JUnit y Mockito
	Práctica de Tema 7 – JUnit y Mockito
13	Trabajo en proyecto
	Trabajo en proyecto

14	Examen
	Trabajo en proyecto
15	Presentaciones proyectos
	Presentaciones proyectos

8. Resultados del aprendizaje

Los resultados del aprendizaje que se esperan del seguimiento de las sesiones de la asignatura y la realización de las prácticas correspondientes, y su relación con los objetivos de la asignatura, son los siguientes:

- Describir qué es la Ingeniería del Software y para qué sirve (CB2, CC16)
- Identificar los distintos modelos proceso software que se pueden aplicar (CG05, CG08, CC16)
- Modelar sistemas software con UML (CG09, CC01, CG01, CC16)
- Identificar las principales arquitecturas software y patrones de diseño (CB2, CB5, CG05, CG08, CC16),
- Aplicar pruebas software para mejorar la calidad de los programas (CB2, CB5, CG05, CC01, CG01, CC16)

De estos resultados del aprendizaje se derivan los criterios de evaluación de estos, que son objeto de la siguiente sección.

9. Procedimiento de evaluación

El procedimiento de evaluación de la asignatura favorece la evaluación continua, que es seguida por un número elevado de alumnos. No obstante, existe la posibilidad de superar la asignatura mediante un examen final.

9.1. Evaluación continua

El alumno puede superar la asignatura realizando una evaluación continua que consiste en pruebas parciales y trabajo en grupo. En este caso no es necesario que realice un examen al final de la asignatura.

- Pruebas parciales (40% del valor de la calificación final). Se realizarán varios exámenes parciales (típicamente 3) en horario de clase a lo largo del curso. Es requisito indispensable para superar la asignatura mediante evaluación continua obtener una calificación igual o superior a cinco en cada una de las pruebas parciales. Estas pruebas parciales constan de ejercicios de dos tipos:
    - *Tests* teórico-prácticos (10% de la calificación final)
    - Ejercicios prácticos en el laboratorio (30% de la calificación final).
- Trabajo en grupo (60% de la calificación final). Consistente en la realización de varias prácticas a lo largo del curso, mediante las cuales se evaluarán los conocimientos adquiridos por el alumno. La individualización de la calificación grupal se realiza mediante encuestas y entrevistas personales en la última semana de clase.

9.2. Evaluación final

Si el alumno ha superado el trabajo en grupo, pero no ha superado alguna de las pruebas parciales, se presentará a un examen reducido, consistente en ejercicios correspondientes a las pruebas parciales no superadas y que, combinado con las calificaciones obtenidas en aquellas que sí ha superado, supondrá el 40% de la nota. El 60% restante corresponderá, como es lógico, a la calificación que haya obtenido en las prácticas.

Si el alumno no ha realizado o no ha superado el trabajo en grupo, se presentará a un examen que supondrá el 100% de la nota, y en el que deberá realizar pruebas teóricas y una práctica de larga duración, de complejidad semejante a las realizadas durante el curso.

9.3. Convocatorias extraordinarias

Este procedimiento de evaluación se aplica tanto en las convocatorias ordinarias como en las extraordinarias. En este último caso, se considerará como nota de prácticas la obtenida en el semestre de docencia de la asignatura al que corresponda la convocatoria.

En el caso especial de alumnos matriculados a tiempo parcial o deportistas de élite, se procederá conforme a la normativa que establece la Universidad de Málaga. Todo ello se establece sin perjuicio de que exista una normativa general sobre evaluación aprobada por el Consejo de Gobierno de la Universidad de Málaga o instancias superiores.

10. Adaptación a modo virtual por COVID o situaciones similares

10.1. Actividades formativas

En caso de que sea preciso adoptar un escenario de docencia semipresencial, las actividades de Grupo Grande (GG) se desarrollarán mediante sesiones virtuales síncronas utilizando herramientas de videoconferencia. Estas actividades tendrán un carácter teórico-práctico sirviendo para presentar tanto los contenidos teóricos de la asignatura como el funcionamiento de las herramientas software recomendadas para realizar las prácticas en grupo. Se considerará la posibilidad de grabar dichas sesiones para ponerlas a disposición del alumnado que por cualquier motivo no haya podido asistir a las mismas. En aquellos aspectos en los que se estime oportuno, se desarrollarán además materiales complementarios, consistentes en notas sobre las presentaciones de diapositivas del curso, vídeos explicativos sobre conceptos y herramientas específicas y/o ejercicios complementarios. En este caso, se grabarían las sesiones con contenidos teórico-prácticos en "píldoras" cuya duración total no debería superar los 25-30 minutos. Los alumnos deberían ver los vídeos antes de asistir a clase. El tiempo en clase estaría destinado a resolver dudas de lo visto en los vídeos y a la realización de ejercicios. Se podría utilizar también la técnica de gamificación para descubrir cuánto han aprendido los alumnos al ver los vídeos en casa y aclarar los conceptos que no hubieran quedado claros.

Por otra parte, en la medida de lo posible las actividades de Grupo Reducido (GR) se realizarán de forma presencial. Estas actividades tendrán un carácter práctico tutorizado por el profesor. Para aquellas actividades de GR que — de acuerdo a las recomendaciones y normas de distanciamiento social de las autoridades sanitarias, a la disponibilidad de los espacios en el centro y a la capacidad docente del profesorado— no pudieran realizarse de forma presencial, se optará por su desarrollo mediante sesiones virtuales síncronas utilizando herramientas de videoconferencia y trabajo asíncrono por parte de los alumnos, para la realización de las iteraciones y las entregas del proyecto en grupo planteado anteriormente para el escenario de docencia presencial.

Si a lo largo del curso fuese necesario adaptarse a un escenario de docencia a distancia o totalmente virtual, las actividades presenciales planificadas se sustituirán por sesiones virtuales síncronas utilizando herramientas de videoconferencia. Como se ha mencionado antes, en este escenario se considera también la aplicación de la metodología de clase invertida.

REFERENCIAS

[1] ACM-IEEE-AIS Computing Curricula 2020 [En línea]. Available: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf [Último acceso: 2022].

[2] Guide to the Software Engineering Body of Knowledge, Versión 3.0, IEEE Computer Society, 2014.

[3] SIE/UPM, «El método del caso. Guías rápidas sobre nuevas tecnologías. Servicio de Innovación Educativa de la Universidad Politécica de Madrid,» 2008. [En línea]. Available: https://innovacioneducativa.upm.es/guias/MdC-guia.pdf. [Último acceso: 2020].

[4] J. Sarramona, *Hacia un nuevo concepto de los programas docentes, Ponencias proyecto innovación educativa, Universidad Politécnica de Valencia,* 1988.

[5] J. Sánchez Núñez, *La técnica expositiva. Instituto de Ciencias de la Educación, Universidad Politécnica de Madrid.,* 2004.

[6] J. Balcázar, Programación Metódica, McGraw Hill, 1993.

[7] D. Knuth, The Art of Computer Programming. Vol. 3: Searching and Sorting., Massachusetts: Addison-Wesley, 1973.