F. J. García-Peñalvo, "Ingeniería del Software," en *Proyecto Docente e Investigador. Catedrático de Universidad. Perfil Docente: Ingeniería del Software y Gobierno de Tecnologías de la Información. Perfil Investigador: Tecnologías del Aprendizaje. Área de Ciencia de la Computación e Inteligencia Artificial* pp. 277-388, Salamanca, España: Departamento de Informática y Automática. Universidad de Salamanca, 2018.

Dr. Francisco José García Peñalvo

Ciencia de la Computación e Inteligencia Artificial

Universidad de Salamanca

Capítulo 7. Ingeniería del Software

One principle problem of educating software engineers is that they will not use a new method until they believe it works and, more importantly, that they will not believe the method will work until they see it for themselves

Watts S. Mumphrey

La incapacidad de las organizaciones para predecir tiempo, esfuerzos y costes en el desarrollo de *software* producido son dos de las principales bases sobre las que surge la *Ingeniería del Software* como una disciplina científica.

Otros factores que contribuyen a la creación de esta disciplina son:

- *Cambios en la relación de costes hardware/software*. Durante los primeros años de desarrollo de las computadoras, el *hardware* sufrió continuos cambios, mientras que el *software* no era considerado sino como un añadido a una máquina, el coste de la cual, además, absorbía la mayor parte del presupuesto destinado a la adquisición del sistema informático. Por ello, en la mayoría de los casos los programas se construían con la finalidad de hacer el uso de las máquinas eficiente, pero en absoluto se pensaba en un desarrollo eficiente de los mismos. Sin embargo, la situación fue cambiando y ya a mediados de los años 60 los costes del *software* ascendieron hasta un 40-50% del coste total del sistema, y su influencia fue creciendo hasta niveles en los que el coste del *hardware* ya representaba tan solo el 20% del total.
- *La importancia creciente del mantenimiento.* La tarea de un desarrollador de *software* no concluye cuando el producto es implementado, instalado y entregado. En efecto, cuando comienza la explotación de un producto *software* se detectan errores, los usuarios en ocasiones piden cambios del mismo, los propios productores pueden facilitar nuevas versiones mejoradas, etc. Todo ello suele considerarse como mantenimiento del *software*. Las tareas de mantenimiento pueden llegar a ser más dificultosas (y, en consecuencia, más caras) que las de desarrollo de un nuevo producto.
- *Los avances en el desarrollo del hardware*. Conforme las generaciones de computadoras se iban sucediendo, los costes de producción de las mismas disminuyeron considerablemente. Esto trajo consigo la implantación casi generalizada del ordenador como herramienta habitual de trabajo, ya que tanto grandes como pequeñas empresas tenían a su alcance algún modelo que cumplía con sus necesidades de proceso de datos a un precio asequible. Evidentemente, el aumento en las ventas de computadores trajo consigo un crecimiento enorme de la demanda de nuevos programas que cubriesen todas las necesidades de los potenciales usuarios.
- *Demanda de software más complejo.* Los avances tecnológicos han propiciado igualmente la aparición de grandes sistemas *software* más complejos que los que pudieran existir en el pasado que resuelven más problemas que los sistemas pequeños. A medida que dichos grandes sistemas se fueron desarrollando, se hizo patente que las técnicas empleadas para desarrollar

pequeños productos (si es que dichas técnicas existían, ya que muchas veces la programación ha sido considerada más arte que ciencia) no funcionaban para aquellos. La necesidad de una aproximación disciplinada al desarrollo de grandes y/o complejos sistemas *software* era, por tanto, evidente. El incremento de la complejidad del *software* no solo se debe al tamaño de los sistemas, también va a influir el aumento de la facilidad de uso de los sistemas *software* y el aumento del número de conexiones en red.

Con el objetivo de vencer todas estas dificultades surgió una nueva disciplina conocida como *Ingeniería de Software*, cuyo nombre se propuso en 1968 en una conferencia de la Organización del Tratado del Atlántico Norte (OTAN) [1] para analizar los problemas del desarrollo de *software*; en esa época había grandes sistemas de *software* que estaban rezagados, que no ofrecían la funcionalidad que requerían los usuarios, que costaban más de lo esperado y que no eran fiables [2]. Como primera aproximación podría tomarse a definición del ANSI/ IEEE Std 729-1983:

> *The systematic approach to the development, operation, maintenance, and retirement of software (La aproximación sistemática al desarrollo, operación, mantenimiento y retirada del software)* [3] (p. 32).

Según lo anterior, el desarrollo del *software* es un problema ingenieril ya que trata de crear soluciones efectivas y viables económicamente hablando a problemas reales.

Antes de entrar a una mayor conceptualización de la *Ingeniería del Software*, conviene dejar completamente clara la definición de *software*, por ejemplo, según el IEEE Std 610.12-1990, se puede definir como:

> *Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system* [4] (p. 66).

O como lo definen Roger S. Pressman y Bruce R. Maxim:

> *Software is: (1) instructions (computer programs) that when executed provide desired features, function, and performance; (2) data structures that enable the programs to adequately manipulate*

> *information, and (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs* [5] (p. 4).

Por su parte Mario G. Piattini y otros definen *software* como:

> *Conjunto de programas, procedimientos y documentación asociada a la operación de un sistema informático* [6, 7].

Los documentos son en la mayoría de los casos tan importantes para el correcto aprovechamiento del *software* como lo es el propio programa fuente. De hecho, hay estimaciones que sitúan al desarrollo de la documentación asociada a los sistemas *software* entre los dos o tres apartados más costosos de todo el proceso [8].

7.1. Definición de Ingeniería del Software

La introducción del término *Ingeniería del Software* se produce en la primera conferencia sobre Ingeniería del Software patrocinada por la OTAN, celebrada en Garmisch (Alemania) en octubre de 1968 [1], no obstante la paternidad del término se le atribuye a Fritz Bauer [9].

Fue tal la aceptación de esta conferencia que se consiguió un nuevo patrocinio de la OTAN para una segunda conferencia sobre Ingeniería del Software, que tendría lugar un año más tarde en Roma (Italia) [10], con unos resultados menos esperanzadores que los producidos en la primera conferencia. De hecho, no se produjo ninguna petición de que continuara la serie de conferencias de la OTAN, lo cual no influyó para que a partir de entonces se utilizara con gran profusión el nuevo término para describir los trabajos realizados, aunque quizás sin un consenso real sobre su significado.

En el contexto educativo, sin duda alguna, lo que más controversia ha levantado es el propio nombre del término, centrándose la discusión en la pregunta *¿es la Ingeniería del Software realmente una Ingeniería?* [11-13].

Los argumentos que se dan para sustentar una respuesta negativa se pueden resumir en dos categorías. La primera reuniría a aquellos que se ciñen a la definición literal de ingeniería dada por algunos diccionarios o sociedades profesionales, argumentando que en estas definiciones se hace mención a productos tangibles derivados del uso efectivo de materiales y fuerzas naturales, mientras que el

software ni es tangible, ni utiliza materiales y/o fuerzas naturales para su concepción. La segunda categoría estaría formada por los que arguyen que una disciplina ingenieril evoluciona desde una profesión y la profesión relacionada con el *software* no ha evolucionado lo suficiente para ser considerada una ingeniería.

Por el contrario, son muchos los que están a favor de la utilización y difusión del término *Ingeniería del Software*, tomando como un estándar de facto la utilización reiterada del término en la bibliografía especializada.

Quizás la defensa más fuerte y adecuada de la Ingeniería del Software como ingeniería venga de la mano de Mary Shaw que justifica que si tradicionalmente se ha definido ingeniería como "la creación de soluciones rentables a problemas prácticos mediante la aplicación del conocimiento científico para la construcción de cosas al servicio de la humanidad" [14], entonces el desarrollo del *software* es un problema ingenieril apropiado, porque involucra "la creación de soluciones rentables económicamente para problemas prácticos" [15].

Una vez hechas estas disquisiciones sobre el término y sus controversias, se va a proceder a exponer una muestra de las numerosas definiciones que de Ingeniería del Software se pueden encontrar en la bibliografía.

> *Ingeniería del software es el establecimiento y uso de principios sólidos de ingeniería, orientados a obtener software económico que sea fiable y trabaje de manera eficiente en máquinas reales.* **Fritz Bauer**, *First NATO Software Engineering Conference, Garmisch (Germany)*, 1968 [16, 17].

> *Multi-person development of multi-version programs*. **David L. Parnas** [18, 19].

> *Software Engineering: The practical application of scientific knowledge in the design and construction of computer programs and the associated documentation required to develop, operate, and maintain them.* **Barry W. Boehm** [20] (p. 1226).

Ingeniería del Software es el estudio de los principios y metodologías para desarrollo y mantenimiento de sistemas software. **M. V. Zelkovitz et al.** [21].

Es la disciplina tecnológica y de gestión que concierne a la producción y mantenimiento sistemático de productos software que son desarrollados y modificados a tiempo y dentro de los costes estimados. **Richard Fairley** [22].

Tratamiento sistemático de todas las fases del ciclo de vida del software. Se refiere a la aplicación de metodologías para el desarrollo del sistema software. **Asociación Española para la Calidad** [23].

La aplicación disciplinada de principios, métodos y herramientas de ingeniería, ciencia y matemáticas para la producción económica de software de calidad. **Watts S. Humphrey** [24].

(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
(2) The study of approaches as in (1).

IEEE Std 610.12-1990 [4] (p. 67).

(1) The systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software.

(2) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

24765-2010 - ISO/IEC/IEEE [25] (p. 331).

Disciplina tecnológica y de gestión concerniente a la invención, producción sistemática y mantenimiento de productos software de alta calidad, desarrollados a tiempo y al mínimo coste. **William B. Frakes et al.** [26].

Aplicación de herramientas, métodos y disciplinas para producir y mantener una solución automatizada de un problema real. **Bruce I. Blum** [27].

Aplicación de principios científicos para la transformación ordenada de un problema en una solución software funcional, así como en el consiguiente mantenimiento del software hasta el final de su vida útil. **Alan M. Davis** [28].

That form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problem. **Watts S. Humphrey** [29].

La aplicación de métodos y conocimiento científico para crear soluciones prácticas y rentables para el diseño, construcción, operación y mantenimiento del software y los productos asociados, al servicio de las personas. **Mary Shaw y David Garlan** [30].

The application of science and mathematics by which the properties of software are made useful to people. **Barry W. Boehm** [31] (p. 12).

Software engineering encompasses a process, a collection of methods (practice) and an array of tools that allow professionals to build high-quality computer software. **Roger S. Pressman y B. R. Maxim** [5] (p. 14).

Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use. **Ian Sommerville** [32] (p. 21).

La Ingeniería del Software requiere la comprensión y aplicación de principios de ingeniería, habilidades de diseño, buenas prácticas de gestión, fundamentos de la Ciencia de la Computación y formalismos matemáticos. Es tarea de la Ingeniería del Software juntar estas áreas de trabajo tan dispares y utilizarlas en las fases de obtención de los requisitos, especificación, diseño, verificación, implementación, prueba, documentación y mantenimiento de sistemas software complejos y de gran tamaño. El ingeniero del software debe cumplir el papel del arquitecto del sistema complejo, tomando en cuenta las necesidades y requisitos del usuario, la viabilidad, el coste, la calidad, la confianza, la seguridad y las restricciones temporales. La necesidad de ajustar la importancia relativa de estos factores de acuerdo a la naturaleza del sistema y de su aplicación confiere una fuerte dimensión ética a las tareas del ingeniero del software, sobre quien pueda depender la seguridad y bienestar de otros, y para quien, como en medicina o en derecho, un sentido de moralidad profesional se requiere para su trabajo.

El ingeniero del software debe ser capaz de estimar el coste y la duración del proceso de desarrollo del software, así como determinar la consecución de corrección y confianza. Tales medidas y estimaciones pueden involucrar conocimientos de conceptos financieros y de gestión, al mismo nivel que el manejo de los fundamentos matemáticos. Se necesita el uso preciso de las notaciones formales y de las palabras para expresarlas con el grado de precisión requerido a otros ingenieros y a clientes formados. En la mayoría de las circunstancias las hebras técnicas, teóricas y de gestión de un proyecto de Ingeniería del Software no pueden separarse unas de las otras.

Para construir grandes productos y conseguir una alta productividad, el ingeniero requiere el uso de herramientas software de desarrollo y

de elementos reutilizables que garanticen su subsiguiente modificación y mantenimiento con seguridad.

La actividad profesional del ingeniero del software abarca el rango de tareas involucradas en el ciclo de vida de un sistema software. La obtención de requisitos, especificación, diseño, verificación y construcción son tareas críticas para conseguir la calidad del producto y son todas ellas responsabilidad del ingeniero del software.

Dado que el software determina el comportamiento de un autómata, el ingeniero del software necesita tener conocimientos de hardware digital y de comunicaciones. Aunque la Ingeniería del Software como disciplina puede ser calificada como independiente del área de aplicación, su realización debe ser en el contexto de aplicaciones específicas. El ingeniero del software debe, por tanto, ser capaz de colaborar con otros profesionales que le brindarán capacidades complementarias en la labor de especificar, diseñar y construir sistemas hardware-software que se ajusten a las necesidades del cliente, haga uso de las soluciones hardware y software en una óptima combinación y ofrezca una interfaz de usuario con una calidad adecuada.

La mayoría del software se construye en equipo, frecuentemente con equipos interdisciplinarios. La habilidad para trabajar cerca los unos de los otros es esencial.

Algunos de los métodos y de las herramientas intelectuales de la Ingeniería del Software están en proceso de desarrollo y se espera que tengan que cambiar de forma rápida. Los ingenieros del software, por tanto, necesitan tener unos buenos fundamentos teóricos que les sirvan de base para aprender y usar nuevos métodos en el futuro, y la mentalidad que les permita actualizar de forma permanente los conocimientos que necesitan para su labor profesional. **The British Computer Society and The Institution of Electrical Engineering** [33].

1. *Definición central:*

- *Ingeniería es la aplicación sistemática de conocimiento científico para la creación y construcción de soluciones rentables a problemas prácticos al servicio de la humanidad.*
- *La Ingeniería del Software es la forma de ingeniería que aplica principios propios de la Ciencia de la Computación y Matemáticas para conseguir soluciones rentables a problemas software.*

2. *Elaboraciones e interpretaciones:*

   - *La creación y construcción del software debe incluir el mantenimiento. Debe cubrirse el ciclo de vida del software completo.*
   - *La rentabilidad implica no solo dinero, sino tiempo, calendario y recursos humanos. También implica obtener buenos valores por los recursos invertidos; lo que incluye la calidad cuando las medidas se consideren oportunas.*
   - *La Ingeniería del Software no se limita a aplicar solo principios de la Ciencia de la Computación y las Matemáticas, sino cualquier principio del que pueda sacar ventaja.*
   - *La Ingeniería del Software necesita contar con principios y técnicas de gestión para llevar a cabo sus actividades de desarrollo.*

3. *Distinción entre el uso actual del término "Ingeniería del Software" y la definición que se adecua a la misión del Software Engineering Institute:*

   - *Actualmente, el término "Ingeniería del Software" tiene múltiples conjuntos de significados conflictivos y pobremente entendidos, que van desde la programación a la gestión del diseño del sistema.*
   - *Actualmente, el término "Ingeniería del Software" es más una aspiración que una descripción.*

Software Engineering Institute [34].

Parece claro que hay un consenso en que el desarrollo del *software* necesita una base rigurosa que encuentra en la Ingeniería, e indirectamente en la Ciencia y en las Matemáticas. Pero concretamente, en lo referente a la palabra *Ingeniería*, hay

diferentes opiniones, pero con el paso del tiempo tienden a consensuarse en que la *Ingeniería del Software* es una disciplina de *Ingeniería*. Por ejemplo, Hoare [35], en la década de los setenta, cita los componentes clave de la ingeniería, que según él son lo suficientemente valiosos y relevantes como para ser imitados por los desarrolladores de *software*, concretamente identifica cuatro aspectos de la ingeniería que cubren tanto los elementos teóricos como los prácticos de la Ingeniería del Software: *profesionalismo*, *vigilancia*, *conocimiento teórico* y *herramientas*; J. A. McDermid [36] destaca los fundamentos de Ciencia y Matemáticas; R. S. Pressman, en ediciones anteriores y en clara alusión a F. Bauer, habla simplemente de principios de ingeniería, que pueden incorporar principios teóricos y prácticos [37], sin embargo, en su última edición pone mucho más énfasis en el proceso y en los métodos, es decir, en los aspectos prácticos, para lograr el fin último, el *software* de calidad, para lo que se presenta la Ingeniería de Software organizada en capas y sustentada en el proceso (ver Figura 7.1), el cual si bien requiere disciplina, necesita también de adaptabilidad y de agilidad [5]; algo similar ocurre con Ian Sommerville que, por ejemplo en la edición de su libro incluye "teorías, métodos y herramientas" [38], aunque indica que la Ingeniería del Software es diferente a otras formas de Ingeniería debido a la propia naturaleza del *software*, sin embargo, en la última edición comienza su definición explícitamente con "La Ingeniería del Software es una disciplina ingenieril" y pone mucho énfasis en que la Ingeniería del Software no solo se refiere a los procesos técnicos para el desarrollo del *software*, sino que además incluye las actividades de gestión del proyecto, así como el desarrollo de herramientas, métodos y teorías para el soporte del desarrollo del *software*; Anthony I. Wasserman sugiere que existen ocho nociones fundamentales en la Ingeniería del Software que forman la base para una disciplina efectiva [39]: abstracción, métodos y notaciones de análisis y diseño, prototipado de la interfaz de usuario, modularidad y arquitectura del *software*, proceso y ciclo de vida, reutilización, métricas, y herramientas y entornos integrados.

Figura 7.1. Capas de la Ingeniería del Software. Fuente: Basado en [5] (p. 16)

No solo hace falta decir lo qué es la Ingeniería del Software, sino que es conveniente recalcar lo qué no es; así la Ingeniería del Software no es el diseño de programas que se implementan en otras áreas ingenieriles, ni es simplemente una forma de programar más organizada que la que prevalece entre aficionados, principiantes o personas con falta de educación y entrenamiento específico.

El concepto de Ingeniería del Software surge de la distinción entre el desarrollo de pequeños proyectos (*programming in the small*) y el desarrollo de grandes proyectos (*programming in the large*), de forma que el reconocimiento de que la Ingeniería del Software está relacionada con esta última. Este primer concepto fue rápidamente ampliado para incorporar a la Ingeniería del Software todas aquellas tareas relacionadas con la automatización de los Sistemas de Información y con la Ingeniería de Sistemas en general.

7.2. Marco conceptual de la Ingeniería del Software

El estudio de las características comunes de los sistemas se conoce como Teoría General de Sistemas [40]. Los principios de esta teoría, derivados del estudio de otros sistemas, se pueden aplicar a los sistemas automatizados. Algunos de los principios generales de la teoría general de sistemas son los siguientes:

- Cuanto más especializado sea un sistema menos capaz es de adaptarse a circunstancias diferentes.
- Cuanto mayor sea el sistema mayor es el número de recursos que deben dedicarse a su mantenimiento diario.
- Los sistemas siempre forman parte de sistemas mayores y siempre pueden dividirse en sistemas menores.