El Problema de Hacer Software
Reflexiones y Profesión

Universidad de Cantabria

Facultad de Ciencias

Francisco Ruiz

Contenidos

- El Problema del Desarrollo de Software
  - Evolución Histórica
  - Nuevos Paradigmas
  - Naturaleza del Problema
- Contexto de la Ingeniería del Software
  - Perspectiva de Ingeniería
  - ¿Por qué Ingeniería del Software?
  - Definición
- Ingeniería del Software vs Informática
  - Currículos Internacionales
  - Mercado Profesional
  - Academia
- ¿Es Hacer Software una Profesión?
- Conclusiones

El desarrollo de software es una rama de la ingeniería muy reciente

El software está en todas partes

“Our civilization runs on software” (Bjarne Stroustrup)

- Se habla de “crisis del software” desde los años 70
  - Da la casualidad que justo los que nos dedicamos a esto somos los peores profesionales, los más “chapuceros”, en todos los países, o
  - Es que nos enfrentamos a un problema difícil, especial y distinto al que se enfrentaron antes otras ingenierías.
- A veces los éxitos se confunden con los fracasos
  - Si somos malos haciendo software, ¿por qué el software es cada vez más frecuente en la vida de cualquier persona y más importante para cualquier organización?
- En realidad, hemos seguido un proceso histórico muy interesante.
  - Para entender donde estamos y hacia donde vamos debemos comprender de donde venimos.

- A lo largo del tiempo hemos sido capaces de resolver una gran cantidad de dificultades, en un camino que siempre se ha caracterizado por:
  - Aprovechar el aumento de potencia y capacidad del hardware para "*hacer software más cerca de las personas y más lejos de las máquinas*".

- Fuimos capaces de trabajar de manera lógica y no física.
  - Los enchufes en clavijas pasaron a ser 0's y 1's.
- Inventamos lenguajes y traductores para poder "representar" mejor los algoritmos como ideas.
  - El código máquina dejó de usarse para programar.
- Ideamos lenguajes "cercanos" al idioma natural o a los idiomas de las ciencias (matemáticas) para mejorar nuestra capacidad de expresar.
  - COBOL se pareció al inglés lo máximo posible.
  - PROLOG se basaba en la lógica matemática.
- Descubrimos que teníamos que organizar bien el flujo de ejecución del código.
  - Programación estructurada (PASCAL).

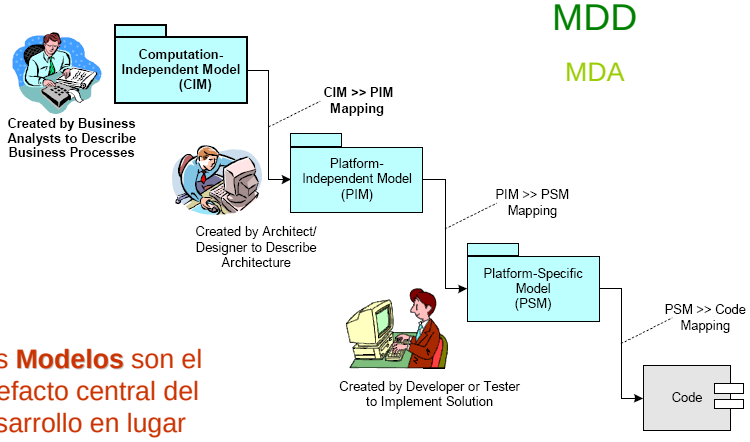
- Conforme el software se fue haciendo más complejo tuvimos que enfrentarnos a nuevos retos:
  - Si tenemos mucho código mejor separarlo en varias partes.
    - Programación modular (MODULA 3).
  - Necesitábamos poder manejar informaciones complejas de distinta naturaleza.
    - Tipos abstractos de datos
    - Sistemas de bases de datos.
  - En un software grande es un lío “organizar” las piezas de código. Necesitamos un criterio para decidir qué piezas tener y qué datos y código poner en cada una.
    - Orientación a objetos.

- Pero seguimos teniendo otros retos pendientes:
  - Si hemos ido subiendo de **nivel de abstracción** en los lenguajes de programación, ¿nos permite la tecnología actual dar otro salto más?.
    - Java es código fuente, y ¿UML no?.
    - ¿Existe alguna manera de construir software más rápidamente y con menos errores?.
  - La **integración** sigue siendo un problema difícil.
    - Integrar sistemas
    - Integrar tecnologías
  - Seguimos teniendo dificultades para **entender** bien a los clientes/usuarios.
    - Muchos proyectos técnicamente correctos fracasan (el software no sirve a los supuestos destinatarios o no lo usan).
  - El software está en la **red**.
    - El concepto clásico cerrado de “aplicación” software está desapareciendo.
    - “Cloud computing”

- Para enfrentar estos retos surgen algunos nuevos paradigmas y plataformas tecnológicas
  - que no son alternativos a los anteriores, sino complementarios
- Desarrollo Dirigido por Modelos (MDD)
  - MDA – Model-Driven Architecture
- Orientación a Servicios (SOC)
  - SOA – Service-oriented Architecture
- Orientación a los Procesos de Negocio (BPM)
  - BPMS – Business Process Management Systems

MDD

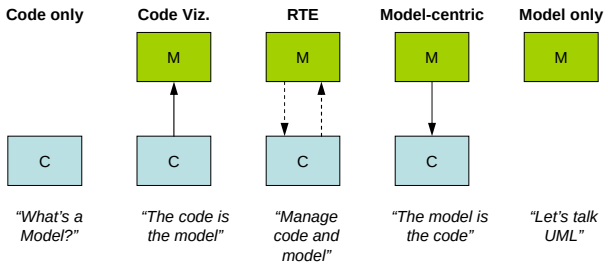
MDA



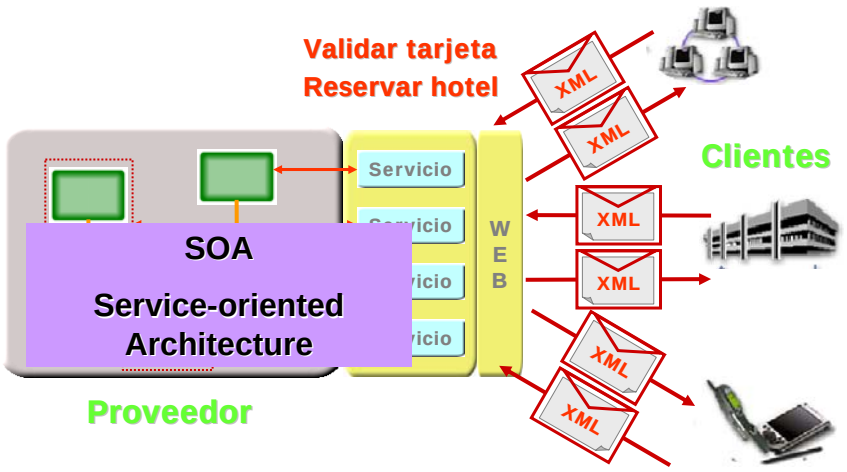
Los **Modelos** son el artefacto central del desarrollo en lugar del **código** fuente

El papel de los modelos

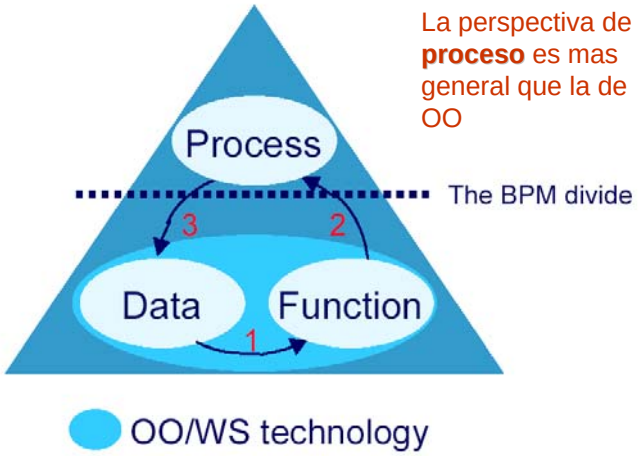
(from: Grady Booch: Strachey Lecture, "The Limits of Software")



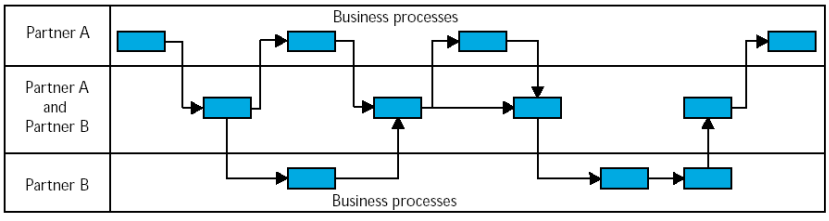
Nuevo enfoque de interacción entre sistemas mediante **servicios**



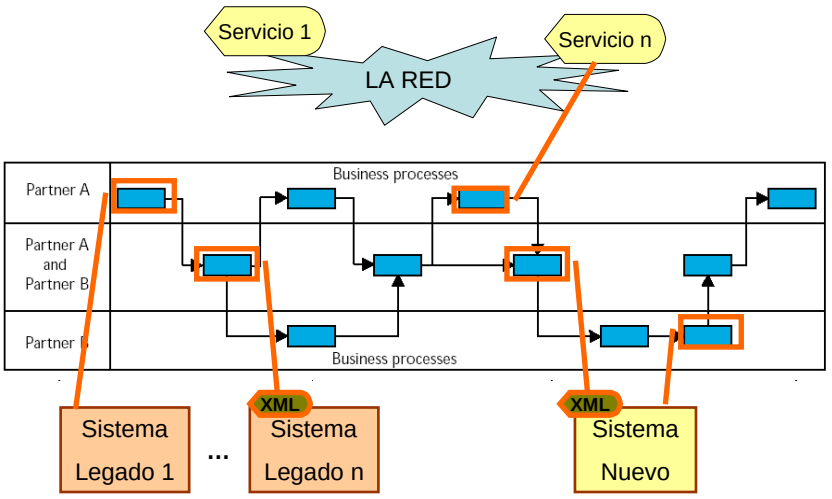
La perspectiva de **proceso** es mas general que la de OO



BPM + SOC + MDD



BPM + SOC + MDD

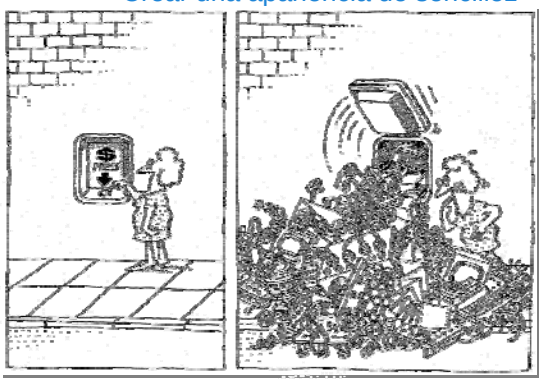


- Booch, G. (2007).
  - The Promise, The Limits, The Beauty of Software.
  - Computer Science Teachers Association, ACM.
  - http://csta.acm.org/Resources/sub/Turing_Lecture.ppt

  - *Software development has been, is, and will remain fundamentally hard.*
  - *It is a tremendous privilege to be a software professional*
  - *It is also a tremendous responsibility*

- Booch, G. (2007).
  - The Promise, The Limits, The Beauty of Software.
  - Computer Science Teachers Association, ACM.
  - http://csta.acm.org/Resources/sub/Turing_Lecture.ppt

Crear una apariencia de sencillez



Ingenio (DRAE)
- Industria, maña y artificio de alguien para conseguir lo que desea.
- Máquina o artificio mecánico (ingenio de azúcar).

Ingeniería (DRAE)
- Estudio y aplicación, por especialistas, de las diversas ramas de la tecnología.

Ingeniero/a
- Persona que aplica los conocimientos de una o varias ramas de la ciencia para resolver cierto tipo de **necesidad** de la gente,
  - Mediante el diseño, construcción u operación de algún tipo de **artefacto o sistema**.

sistema

proceso

INGENIERO/A

palabras clave

proyecto

usuario

- Cualquier ingeniería se caracteriza porque:
  - Se necesitan conocimientos avanzados para diseñar y construir el tipo de sistemas que la caracteriza.
    - Diferencia entre técnico e ingeniero.
  - Existen dos “momentos”:
    - Primero, conocer el problema, y
    - Sólo después, podemos diseñar y construir la solución.
  - Para conseguir buenos resultados (en calidad, tiempo y costes) es necesario trabajar de forma organizada y sistemática.
  - La creatividad es necesaria (diseño), pero no es suficiente,
    - Diferencia entre artista e ingeniero.

El **sentido común** es muy importante.

- Ley del Mínimo Esfuerzo
  - Entre las opciones correctas elegir la más sencilla.
  - Reutilización
    - Del código, del resto de artefactos software y del conocimiento.
- No inventar la rueda
  - Emplear estándares.
- Zapatero a tus zapatos
  - No linealidad entre escala vs complejidad.
- Aprender de la experiencia (nuestra o de otros).
  - Utilizar “buenas prácticas” y “lecciones aprendidas”.
  - Seguir la “Ley de Murphy”

- La ingeniería existe porque las personas diseñan y construyen artefactos/sistemas cada vez más complejos.
- El mayor nivel de complejidad que el ser humano ha enfrentado a lo largo de su historia se encuentra en algunos de los sistemas software actuales
  - Sistemas operativos como Windows Vista, Linux
  - Coches, aviones (ver http://spectrum.ieee.org/green-tech/advanced-cars/this-car-runs-on-code)
- Un indicador de la complejidad de un sistema es el número de variables independientes que afectan al comportamiento del sistema.
  - En un sistema físico (edificio) son decenas o cientos.
  - En un sistema software (red de telecomunicaciones, control de tráfico aéreo) pueden ser miles o decenas de miles.

- ¿A qué se parece el software?
  - A un frigorífico (que se fabrica).
  - A un libro (que se idea y se escribe).
  - A una receta de cocina (que se inventa y se anota).
  - A un servicio de un abogado en un juicio (que nos ayuda con su conocimiento especializado).
- ¿Producto o Servicio?.
- Entonces, ¿la gente que hace software qué clase de habilidades y capacidades debe tener?
  - Arquitecto
  - Albañil
  - Jardinero
  - Artista

Definición

Aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación (funcionamiento) y mantenimiento del software; es decir, la aplicación de los principios y hábitos de la ingeniería al software.

(IEEE,1993)

ACM distingue 5 currículos diferentes

- Ciencia de la Computación (*Computer Science*)
- Ingeniería de Computadores (*Computer Engineering*)
- **Ingeniería del Software (*Software Engineering*)**
- Sistemas de Información (*Information Systems*)
- Tecnología de la Información (*Information Technology*)

¿Por qué los distingue?

¿Es que Informática no es una profesión, y sí lo es Ingeniería del Software?

ACM Computing Curricula 2005. Overview Report

- Pesos asignados a los tópicos de Ingeniería del Software

Área de Conocimiento	CE		CS		IS		IT		SE	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
Fundamentos de Programación	4	4	4	5	2	4	2	4	5	5
. . . . .										
Desarrollo de Sistemas de Información	0	2	0	2	5	5	1	3	2	4
Análisis de Requisitos Técnicos	2	5	2	4	2	4	3	5	3	5
Fundamentos de Ingeniería para Software	1	2	1	2	1	1	0	0	2	5
Economía de Ingeniería para Software	1	3	0	1	1	2	0	1	2	3
Modelado y Análisis de Software	1	3	2	3	3	3	1	3	4	5
Diseño de software	2	4	3	5	1	3	1	2	5	5
Verificación y Validación de Software	1	3	1	2	1	2	1	2	4	5
Mantenimiento del Software	1	3	1	1	1	2	1	2	2	4
Procesos Software	1	1	1	2	1	2	1	1	2	5
Calidad del Software	1	2	1	2	1	2	1	2	2	4
. . . . .										
Desarrollo Digital	0	2	0	1	1	2	3	5	0	1
. . . . .										

Papel Social del Informático (Dahlbom & Mathiassen, 1997):

	Construyo cosas	Ayudo a la gente	Cambio las cosas
Orientación	Máquina	Cultura	Poder
Actividad	Construcción	Evolución	Intervención
Papel	Ingeniero	Facilitador	Emancipador

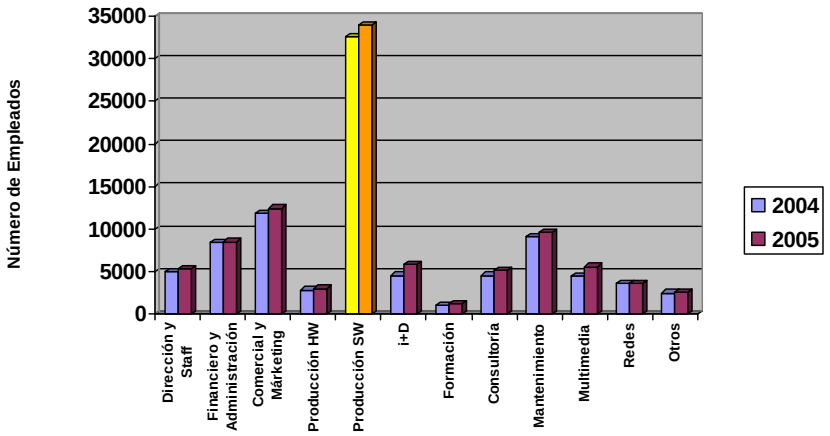
Esto implica tres tipos de roles diferentes dentro de la informática:

- los ingenieros de **desarrollo**: construyen artefactos software o hardware;
- el personal de **soporte**: ayuda a usar aquello que otros desarrollaron; y
- los **consultores**.

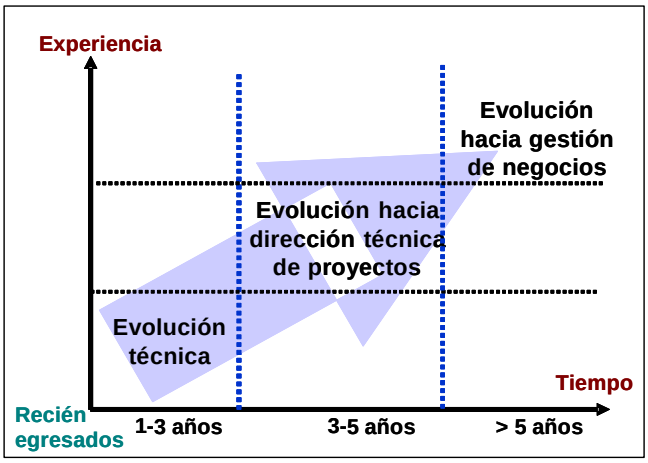
Distribución de los ocupados en perfiles TIC en la Unión Europea 15 (miles), *Career-Space.*

Ocupaciones (SOC90)	Total Puestos TIC	% Incr. 2000-2004
Analistas y Programadores	1.885	+6,1
Ingenieros de Software	1.306	+10,0
Administradores de Sistemas Informáticos	1.019	+4,1
Operadores Informáticos	696	-0,5
Consultores y Gestores	437	+3,7
Ingenieros de Diseño y Desarrollo TIC	399	+0,2
Ingenieros de Computadores	348	+6,5
Ingenieros Eléctricos	203	-0,5
Ingenieros Electrónicos	196	+3,0
Total TIC	6.489	+4,7
Total Empleo	166.696	+0,8

Telefonica (2007): Personal por áreas funcionales en el sector de TI en España).



Informe PAFET (2002): Evolución profesional habitual de los profesionales TIC.



Características de una Profesión:

- Campo duradero de preocupación/interés humano.
- Cuerpo de conocimientos codificado
  - Conocimiento conceptual
- Cuerpo de prácticas codificado
  - Conocimiento experimental
- Estándares de competencia, ética y práctica
  - Responsabilidad profesional

¿Cómo está la Informática?

- *P. Denning. El Futuro de la Profesión de TI. Novática, nº 147.*
  - Aprender de otros campos ya consolidados.
    - MEDICINA vs SALUD.
    - ABOGACÍA vs DERECHO.
- En Informática todavía se confunden tres cosas diferentes:
  - Sector Económico – Profesión – Puesto de Trabajo

    Salud Médico Cirujano
- Un título académico forma para una o varias profesiones dentro de un cierto sector económico.

- ¿Cuál de las tres cosas es Informática?
  - Sector Económico – Profesión – Puesto de Trabajo

¿?	¿?	Informática
¿?	Informática	Ing. Software
Informática	Ing. Software	Analista

- ¿Y Hacer Software?
  - Construir edificios no es una profesión. La profesión es arquitecto, albañil.
  - Profesiones relacionadas con Hacer Software:
    - Ingeniero de Software
    - Programador
    - ....

- Hacer Software es un problema complejo y seguirá siéndolo.
- La Ingeniería del Software pretende resolverlo mediante la aplicación de maneras sistemáticas y metódicas de trabajar (igual que hicieron hace tiempo otras ingenierías).
- Existe una creciente opinión internacional para que se separe de la Informática tradicional (Ciencia de la Computación).
- Es vital para el futuro (profesional, laboral y académico) de la Informática que se incida más en la perspectiva de ingeniería.
  - Más arquitecto, menos albañil.
- La carrera de Ingeniería Informática prepara para los trabajos más cualificados dentro de un sector económico, que está llamado a tener varias profesiones diferenciadas.
- Una de las profesiones será ingeniería de software.