

Diseño de Software: ¿Ciencia, Ingeniería o Arte?

Por Jesús Zavala

Profesor de la Fundación Arturo Rosenblueth / Est. Org. UAM-I

Congreso Nacional de Software Libre 2005

Universidad Autónoma Metropolitana

jzavalar@yahoo.com <http://www.angelfire.com/scifi/jzavalar>

México, D.F. 22-25/feb/2005

¿Por qué estudiar el diseño?

- ¿Podemos *diseñar mejor* enfocándonos en el proceso de diseño?
- ¿Podemos *enseñar a otros* el diseño?
- ¿Podemos *organizar y administrar* el diseño?
- ¿El diseño es ciencia, ingeniería o arte?

- Analicemos la situación ...

Agenda

1. El software y su importancia
2. La ciencia. En busca de los principios
3. La ingeniería. La mejor forma
4. El arte. La anarquía organizada
5. El diseño de software. ¿Es posible?
6. Conclusiones
7. Referencias

1. ¿Qué es software?

- Software es “la suma total de los programas de cómputo, procedimientos, reglas y documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo” [IEEE 1995]
- Lo invisible y etéreo
- “El alma de la compu”
- “El coco” de los “analfabetos tecnológicos”



Importancia del software

- El gran invento del siglo XX... Turing, Von Neumann
- Es omnipresente, está en todo: la lavadora, el tostador, el *walkman*, la TV, el despertador, el juguete, el banco, la tienda, etc. y todavía falta más...
- ¿Es el nuevo “big brother”? (¿Que todo lo vé y lo sabe?)
- El activo más importante de la organización moderna (“*Monsters Inc*”)
- El medio para la comunicación y coordinación más potente (Internet / intranet / extranet, “*Voice over IP*”)
- El motor de la “nueva” economía, el comercio electrónico, la “*Webonomics*” (*conocimiento = poder*)

El poder del software

- La automatización
- Pero... también es...
- La esperanza de mejorar la productividad
 - La punta de lanza de la Reingeniería y la TQM
 - El enemigo de la fuerza laboral
 - La amenaza de las filas obreras de despido
 - El pretexto para los despidos masivos por reemplazo
 - La base del “nuevo” paradigma económico

La crisis del software

- El cohete Ariane 5 de la Agencia Espacial Europea se autodestruyó
 - Una falla en el sistema de control de rastreo (varios millones en pérdidas)
- La NASA en 1999 pierde un satélite en Marte
 - La conversión incorrecta entre unidades inglesas y métricas
- El Aeropuerto de Denver se retrasó
 - Retraso de la apertura del aeropuerto 16 meses (\$1 millón dólares/día)
- El Proyecto Génesis *Department of Motor Vehicles (DMV)* del estado de Nevada EU
 - Trastocó todos los órdenes de poder en la organización intentando cambios en los sistemas, la estructura, la gente y la cultura
 - El costo del proyecto se elevó de \$34 a \$173 millones de dólares
 - El retraso del servicio aumentó de 40 minutos a 7 horas
- En EU en 1995, se destinaron 81 mil millones de dólares por año a proyectos de desarrollo de software (Standish Group 1995)

Zavala (2004)

El desarrollo de software real 1

- La práctica profesional
 - Ni ciencia, ni ingeniería, ni arte
 - Una práctica *ad-hoc*
 - Certificación: ilusión de calidad y un gran negocio
- Los procesos de desarrollo
 - Rigurosos y laxos a la vez
 - No existen los mejores procesos
- Los desarrolladores
 - Talentosos y con habilidades pero pocos
 - Todólogos: importa el costo no las habilidades
 - Sin planeación de carrera
 - Ni mentoría: no hay transferencia de conocimiento
 - Sobreexplotados (12-16 hr/día) (Ben Ari:2004)
 - Sometidos por el *management: La cultura de la excelencia* (Peters y Waterman 1985)

El desarrollo de software real 2

- La industria del software
 - Muy lejos de convertirse en una industria madura
 - Altos índices de fracasos y promesas incumplidas
 - Algo de *know-why (el saber por qué, ciencia)*
- El conocimiento
 - Las mejores prácticas son aquellas que se adaptan a las particularidades de la empresa de software
 - No se aprovechan las capacidades de la gente
 - Visión cortoplacista de corte económico (Ben Ari:2004)
 - No hay transferencia de habilidades
 - Imposible hacer bases de conocimiento tácito:
 - *know-what – el saber qué*
 - *know-how – el saber hacer*
 - *know-who – el que sabe*

2. La ciencia. En busca de los principios

- Conocimiento exacto y razonado de ciertas cosas
- Conjunto de conocimientos fundados en el estudio
Pequeño Larousse
- Proceso sistemático para entender cómo trabaja el mundo natural
- Ejemplos:
 - Ciencias naturales
 - Ciencias sociales
 - Ciencias de la computación
- Basada en: la filosofía, la lógica, la epistemología y la ética

La ciencia. Sus visiones

- La búsqueda de “la verdad”
- Generalmente ignora los intereses del investigador, aunque en su visión posmodernista no
- El pensamiento científico y su dualidad (¿ambos?)
 - Deductivo vs inductivo
 - Racional vs irracional
 - Objetivo vs subjetivo
 - Positivista vs constructivista
 - etc.

La ciencia. Su método

- Ciencia = “investigación científica a través del modelado y simulación de procesos físicos en computadoras”
- “La ciencia, tradicionalmente se ha visto como un paradigma para descubrir las leyes de la naturaleza”
- “el paradigma consiste en formular una hipótesis, realizar predicciones basadas en la hipótesis, recolectar datos y analizar los datos para confirmar o rechazar la hipótesis”
(Denning 1999:31)

La ciencia. Sus atributos

- “Atribuirle perfección al conocimiento científico, es además de una falsedad, una torpeza ya que implica quitarle su mayor mérito y su principal atractivo: ser obra, exclusivamente, del ser humano...
- Lo característico es su perpetuo interrogante respecto de ella misma...
- Todas las formas de conocimiento científico están apenas brotando”

Fregoso (1988:84)

La ciencia. ¿Ciencias de la computación?

- “Es el cuerpo de conocimiento que involucra el diseño, análisis, implementación, eficiencia y aplicación de los procesos que transforman información. La pregunta fundamental común a toda la ciencia de la computación es ‘¿qué se puede automatizar (eficientemente)?’”

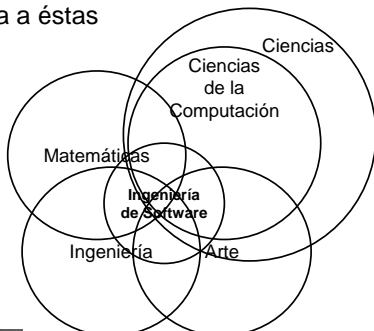
Denning (1985:16)

- La base de la ingeniería de software
- Los principios básicos
- Basada en tres paradigmas:
 - Ciencia: leyes y principios de la naturaleza (?)
 - Matemáticas: algoritmos, álgebra booleana, etc.
 - Ingeniería: El diseño de ingeniería y aplicación práctica

Denning (1999:29)

Las ciencias de la computación. ¿Base del software?

- Una ciencia basada en otras ciencias que se aplica a éstas



Las ciencias sociales. ¿El eslabón perdido?

- Conocimiento científico enfocado al estudio del ser humano como ser social
- Estudian la realidad social: al hombre, los grupos y las sociedades
- La visión *soft* no presente en el software
- La predicción no es su mejor atributo
- Apertura a todas las posibilidades
- Convergencia y divergencia a la vez y crítica constante
- Ejemplos:
 - Antropología, sociología de las organizaciones
 - Economía, administración
 - Estudios organizacionales: estrategia, decisiones...
 - Teoría de la organización: ¿aporte a la ing de requerimientos?

La Ciencia. Resumen

- Las ciencias sean exactas, naturales o sociales se centran en la búsqueda de los principios que describen y explican a sus respectivos objetos de estudio
- Utiliza el método científico como guía para plantear y resolver sus hipótesis y plantear teorías
- Pretende ser “la verdad” que lo explica, independientemente del sujeto que la estudia
- Las ciencias sociales son menos predictivas por centrarse en el estudio del ser humano en sociedad, pero permiten la amplia exploración filosófica, metodológica y teórica
 - Modelos de organización
 - Estrategia

3. La ingeniería. La mejor forma ante todo

- Aplicación de las ciencias... a la invención, perfeccionamiento y utilización...
 - Conjunto de estudios que permiten determinar... las orientaciones más deseables, la mejor concepción, las condiciones... óptimas y los materiales y procedimientos más adecuados
- Pequeño Larousse
- Ingeniería civil
 - Ingeniería electrónica
 - Ingeniería de software... ¿es predecible y única?
- Basada en las ciencias
 - La mejor forma de hacer las cosas:
 - *one-best-way* (F. W. Taylor) OCT

¿Cómo piensan los ingenieros el diseño?

- Objetivo UNTIL (diseño "suficientemente bueno") o (se termine el tiempo)
 - Desiderata (lista de deseos) DO otro diseño (mejorar la función de utilidad)
 - Función de utilidad UNTIL diseño esté completo
 - Restricciones, especialmente presupuestales (no necesariamente costo \$) WHILE diseño sea factible, tomar otra decisión de diseño
 - Diseño del árbol de decisiones ENDO WHILE
 - Tomar el mejor diseño Recorrer el árbol de decisiones
 - Veamos algo más familiar... Explorar un camino no buscado antes
- END UNTIL
END DO
Tomar el mejor diseño
END UNTIL

Brooks (1995), Brooks (2000:7)

Lo equivocado de este modelo 1

- Realmente NO conocemos el objetivo al principio
 - La parte más difícil del diseño es decidir *qué* diseñar
 - La función más importante del diseñador es ayudar al cliente a decidir lo que quiere
- Aquí es donde están equivocados los expertos
 - Se pierde la visión fresca
 - La visión no es suficiente
 - No se diseña, sino crece conforme aparecen las necesidades
- La *desiderata* y sus prioridades permanecen cambiantes
- Usualmente no conocemos el árbol de decisiones
 - La racionalidad limitada (H. Simon)
 - Las decisiones estratégicas basadas en información: ¿son una falacia? (Mintzberg 1973)
- Las restricciones permanecen cambiantes
 - El mundo siempre cambia
 - La única constante: el cambio

Brooks (2000:11-19)

Lo equivocado de este modelo 2

- El modelo racional está equivocado porque no describe lo que realmente sucede
 - Domina en la literatura de ingeniería
- Los diseñadores más expertos no trabajan de esa manera
 - Ubisoft - Canadá
- Puede dar resultados estrafalarios
 - Las especificaciones funcionales del helicóptero LHX
- Ese modelo lo tenemos en la Ingeniería de Software
- Cuando se compra de un nuevo edificio o un avión:
 - Se paga por una fase de diseño, se aprueba un diseño y se contrata para su implementación
 - Un constructor paga por una fase de diseño y vende las implementaciones

Brooks (2000:11-19)

4: El arte. El caos y la creatividad

- Conjunto de reglas para hacer bien una cosa
- Conjunto de reglas de una profesión
- Habilidad, talento, destreza
- Conocimiento verificable, racional y práctico
- A través de la técnica... busca la distracción y lo estético

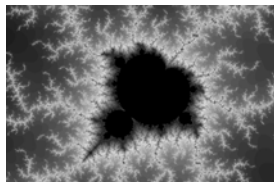
Pequeño Larousse
- Se apoya en las ciencias y crea su propia visión...
- Muchos caminos, ¿todos válidos? No al "one-best-way"...
 - Las bellas artes: música, la pintura...
 - La cuarta división: Ciencias y Artes para el Diseño: Una aportación de la UAM-X al mundo; Arquitectura
 - La guerra: táctica, estrategia y el papel del actor...
 - El diseño de software: Cada nuevo proyecto es distinto;
 - Ubisoft video-juegos: Creatividad y acción sin limitaciones...

El arte. El caos

- ¿o anarquía? Caos con orden...

Un artista:

- nunca se equivoca
 - Tiene accidentes felices (B. Ross)
- no tiene límites
 - Solo su imaginación
- crea con y a pesar de las herramientas y materiales
- trabaja con pasión



5. Diseño

- "Formar un plan o esquema, disponer o concebir en la mente... para su ejecución posterior" (*visión del management*)

Oxford English Dictionary
- "Actividad creativa ... determinar las propiedades formales o las características... de los objetos que se van a producir artística o industrialmente"

Pequeño Larousse

Diseño de software. ¿Es posible?

- La ingeniería de software
 - está en crisis; un fracaso de enfoque, método o fundamentos
 - es un modelo agotado
- El diseño
 - imposible sin liberarlo de sus ataduras: sus metodologías rigurosas y su supuesta predicibilidad
 - se semeja más a las ciencias sociales y humanidades
- El análisis y diseño de sistemas
 - los aspectos más críticos (el qué y el cómo)
 - imposible sin la abstracción de la forma de organización
 - interdisciplinario
- Los proyectos de software libre (FLOSS)
 - anarquías organizadas, las organizaciones posmodernas
 - muchos operan más por 'fuerza bruta' que por diseño
 - falta documentación y diseño
 - falta aplicar las ciencias de la computación, no solo programación: teoría de bases de datos y *usability*

El diseño; ¿ciencia, ingeniería o arte? 1

- Ciencia
 - Las bases de las ciencias donde se aplica el diseño
 - El método científico y filosofía de la ciencia
 - Visión positivista: la verdad
 - Manejo político de los descubrimientos
 - Visionaria, humana y tendenciosa
- Ingeniería
 - La aplicación práctica de la ciencia al diseño
 - La destreza y habilidad en la creación
 - La mejor forma de hacer las cosas (*one-best-way*)
 - Pregona el seguimiento riguroso del plan: burocracia
 - El ciclo de vida en cascada está agotado
 - Serios problemas de abstracción en la fase de análisis y diseño por el modo de pensamiento lineal, matemático
 - Un modelo agotado que no explica cabalmente la realidad

El diseño; ¿ciencia, ingeniería o arte? 2

- Arte
 - El uso de las ciencias y de la ingeniería en el diseño: La cuarta división (CyAD)
 - La libertad de elección y exploración de caminos sin limitaciones
 - Importa el objeto, pero también el sujeto: el juego y el jugador
 - Visión constructivista crítica al positivismo
 - Impredescible
 - Niega la existencia del *one-best-way*; muchas mejores formas
 - Entre el caos y la anarquía surge la creatividad
 - La regla es que no hay reglas: creación de videojuegos
- El diseño de software no es ciencia, tampoco ingeniería; es arte que se apoya en las ciencias, incluso en las ciencias de la computación**
- La profesión: diseñador de software**

La Catedral y el Bazar. Modelo evolutivo

- Los brillantes ensayos de Raymond sobre el proceso de Linux
- El Bazar es un modelo evolutivo; ¿supera al modelo en espiral?
- No hay comité de diseño, cada pieza tiene integridad conceptual
- Enfatiza las pruebas pronto y a gran escala
- Reúne muchas mentes para corregir, no para probar
- Se basa en la cultura del talento y la cultura de prestigio
- Es una federación de personas alejadas
- Trabaja cuando los constructores son los clientes
- Se conocen los requerimientos a partir de la experiencia personal

Brooks (2000:26)

¿Cómo crear grandes diseñadores?

- Debemos propiciar su crecimiento deliberadamente
 - Reclutar mentes brillantes para el diseño, no con capacidades de hablar
 - Hacer la escalera dual real y honorable
 - Planear la carrera y mentoría, tal como a los *managers* (como en la India) (Pathibandla y Petersen 2002)
 - Experiencias planeadas, estudios y rotaciones
- Debemos *manejarlos* con imaginación
 - Una 'fábrica de software' humana
 - Solo los seres humanos crean y generan riqueza
- Debemos protegerlos ferozmente
 - de lo *manager*
 - de la gestión

Brooks (2000:43-46)

Creciendo uno mismo como diseñador

Sugerencias:

- Diseñar cantidad de cosas y conservar un cuaderno de notas
 - Las libretas de Da Vinci
- Reflejarlo al escribir sobre tus propias experiencias
 - Utilizar la imaginación, no solo la computadora
 - Escribir, dibujar, esquematizar, etc...
- Estudiar otros diseños documentados
 - Escribir revisiones de herramientas, software, videojuegos, etc.
- Lo que importa es el conocimiento
 - El activo más importante

Brooks (2000:47-51)

6. Conclusiones

- Las ciencias de la computación son parte de los fundamentos del diseño; las otras son las ciencias y artes para el diseño y las ciencias sociales y humanidades
- La crisis del software confirma que el modelo de desarrollo de la ingeniería de software está agotado
- No existe el mejor diseño, sino muchos diseños satisfactorios
- El diseño de software es más complejo de lo que comúnmente nos imaginamos
- El diseño es posible si se libera el pensamiento racional de las restricciones
- El diseño de software solo es posible mediante un ciclo de vida evolutivo, por aproximaciones sucesivas
- La única constante en el diseño es el cambio permanente

7. Referencias 1

- Ben-Ari, Mordechai (2004) "The Software Factory", DSTW, Institute of Science Rehovot, Israel. Disponible en: <http://stwww.weizmann.ac.il/G-CS/BENARI/articles/factory.pdf> (12/04/04)
- Brooks, Frederick P., Jr. (1995) *The Mythical Man-Month*, Addison-Wesley, USA. (el libro indispensable)
- ----- (2000) "The Design of Design" *SIGGRAPH 2000*, jul 25, 2000. Disponible en: <http://terra.cs.nps.navy.mil/DistanceEducation/online.siggraph.org/2001/SpecialSessions/2001TurinoLecture-DesignOfDesign/FredBrooksTuringLecture25July2000.ppt> (10/01/05)
- Denning, Peter. (1985) "The Science of Computing. What is Computer Science?". *American Scientist*. Vol. 73. Jan-Feb 1985. pp. 16-19.
- ----- (1999) "Computing the Profession" en Greening, Tony (Ed.) *Computer Science Education in the 21st Century*, Springer, USA. pp. 27-46
- Fregoso Urbina, Arturo (1988) *Universidad y Vida*. Trillas, México.
- García-Pelayo y Gross, Ramón (1993) *Pequeño Larousse Ilustrado*. Ediciones Larousse, México.

Referencias 2

- IEEE (1990) *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std 610.12-1990, Institute of Electrical and Electronic Engineers, Inc., New York, NY, December 10, 1990
- Mintzberg, Henry (1973) *The Nature of Managerial Work*, Haper & rou, New york, 298 p.
- Peters, Thomas J. y Watermann, Robert A. (1984) *En Busca de la Excelencia*, México, Lasser, 351 p.
- Simon, Herbert Alexander (1977) *The New Science of Management Decision*, Englewood Cliffs- Prentice Hall, New Jersey, 175 p.
- Standish Group (1995) "The Chaos Report", Disponible en: http://www.standishgroup.com/sample_research/chaos_1994_1.php (15/nov/2003)
- Zavala Ruiz, Jesús (2004) "¿Por qué Fracasan los Proyectos de Software?; Un Enfoque Organizacional", Consol 2004. Disponible en: <http://www.consol.org.mx/2004/material/63/por-que-fallan-los-proy-de-soft.pdf> (12/1/2005)

Zzzz! Zzzzz! Zzzz!

Muchas gracias!
¿Preguntas?

Jesús Zavala

jzavalar@yahoo.com

<http://www.angelfire.com/scifi/jzavalar/>