Ingeniería de Software

Universidad Tecnológica Nacional
Facultad Regional Córdoba
Cátedra de Ingeniería de Software
Docentes: Judith Meles – Laura Covaro

Definición de Software

Software, en general, es un set de programas y la documentación que acompaña.

- Existen tres tipos básicos de software. Estos son:
  - System software
  - Utilitarios
  - Software de Aplicación

Margaret Hamilton, lead software engineer, Project Apollo. Mostrando su código fuente..
https://medium.com/@3fingeredfox/margaret-hamilton-lead-software-engineer-project-apollo-158754170da8

5 Razones para no comparar software y manufactura

✓El software es menos predecible

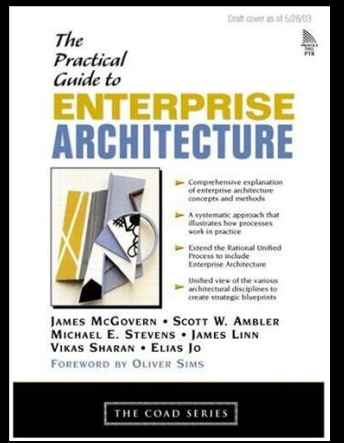
✓No hay producción en masa, casi ningún producto de software es igual a otro.

✓No todas las fallas son errores

✓El software no se gasta

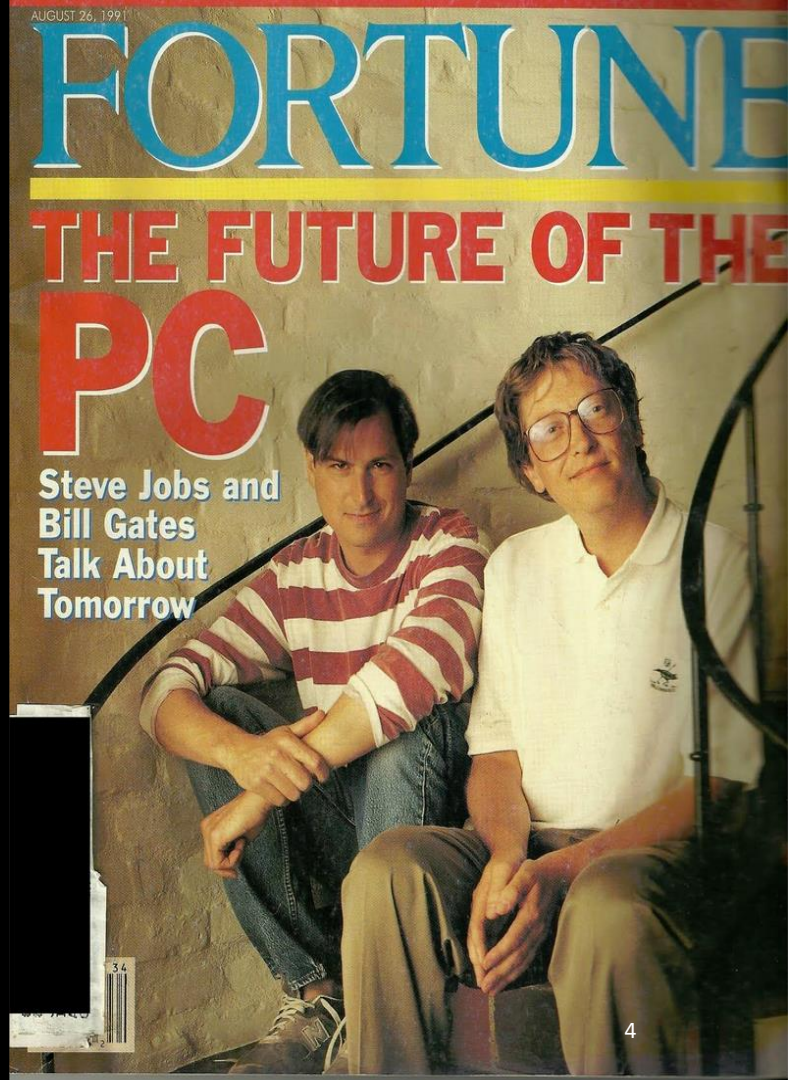
✓El software no esta gobernado por las leyes de la física ☺

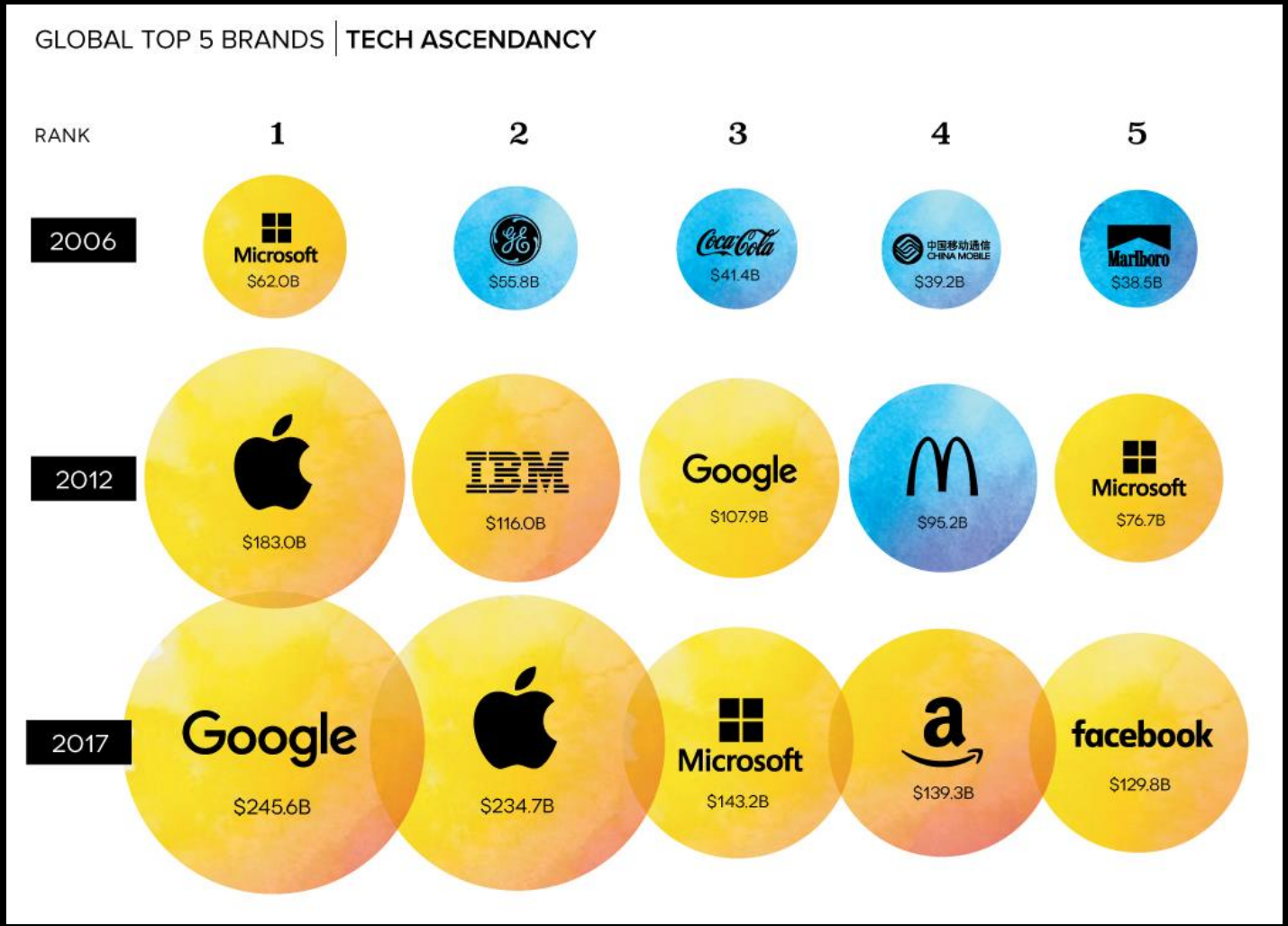
"The creation of software is an intellectual human endeavor. Creating good software relies on the personalities and the intellects of the members of the teams that create it. When applied to a different team of developers a process that delivers great software for one team of developers may fail to deliver anything at all for another team."
-- *The Practical Guide to S/W Arch.*



Un poco de historia

- 1968 • Nace el termino – conferencia de la NATO
- 1975 • The Mythical Man-Month – Frederick Brooks
- 1978 • Tom DeMarco introduce Structured Analysis
- 80' • Primeros grandes errores de software conocidos
- 1987 • No silver bullet (Brooks). Características esenciales del software
- 1989 • Managing the Software Process – Watts Humphrey
- 1990 • Internet / Object Oriented
- 1991 • CMM 1.0
- 1993 • CMM 1.1
- 2000 • CMMI 1.0
- 2001 • Agile Manifiesto
- 2003 • Lean Software Development




GLOBAL TOP 5 BRANDS | TECH ASCENDANCY
RANK
1
2
3
4
5
2006
Microsoft
$62.0B
$55.8B
Coca-Cola
$41.4B
中国移动通信
CHINA MOBILE
$39.2B
Marlboro
$38.5B
2012
$183.0B
IBM
$116.0B
Google
$107.9B
$95.2B
Microsoft
$76.7B
2017
Google
$245.6B
$234.7B
Microsoft
$143.2B
$139.3B
facebook
$129.8B

Las 10 empresas más grandes del mundo en 2019

1. Apple (881.980 millones de dólares)
2. Microsoft (803.090 millones de dólares)
3. Amazon (739.460 millones de dólares)
4. Alphabet (711.940 millones de dólares)
5. Berkshire Hathaway (536.530 millones de dólares)
6. Johnson & Johnson (396.210 millones de dólares)
7. Alibaba (387.610 millones de dólares)
8. Facebook (378.050 millones de dólares)
9. JPMorgan Chase & Co (368.560 millones de dólares)
10. Tencent Holdings (353.670 millones de dólares)

https://www.ig.com/es/estrategias-de-trading/las-10-empresas-mas-grandes-del-mundo-por-capitalizacion-bursati-190124

Algunos problemitas con el desarrollo de software

La versión final del producto no satisface las necesidades del cliente.

No es fácil extenderlo y/o adaptarlo. Agregar más funcional en otra versión es casi una misión imposible

Mala documentación

Mala calidad.

Más tiempos y costos que los presupuestados

Errores informáticos más costosos de la historia

https://www.javiergarzas.com/2013/05/top-7-de-errores-informaticos.html

1 – La destrucción del Mariner I (1962). 18,5 millones de dólares.

El del Mariner I, una sonda espacial que se dirigía a Venus, se desvió de la trayectoria de vuelo prevista poco después del lanzamiento. Desde control se destruyó la sonda a los 293 segundos del despegue. La causa fue una fórmula manuscrita que se programó incorrectamente.

2 – La catástrofe del Hartford Coliseum (1978). 70 millones de dólares.

Apenas unas horas después de que miles de aficionados abandonaron el Hartford Coliseum, el techo se derrumbó por el peso de la nieve. La causa: calculo incorrecto introducido en el software CAD utilizado para diseñar el coliseo.

3 – El gusano de Morris (1988). 100 millones de dólares.

El estudiante de posgrado Robert Tappan Morris fue condenado por el primer ataque con "gusanos" a gran escala en Internet. Los costos de limpiar el desastre se cifran en 100 millones de dólares. Morris, es hoy profesor en MIT.

4 – Error de cálculo de Intel (1994). 475 millones de dólares.

Un profesor de matemáticas descubrió y difundió que había un fallo en el procesador Pentium de Intel. La sustitución de chips costó a Intel 475 millones.

5 – Explosión del cohete Arian (1996). 500 millones de dólares.

En el 1996, el cohete Ariane 5 de la Agencia Espacial Europea estalló. El Ariane explotó porque un número real de 64 bits (coma flotante) relacionado con la velocidad se convirtió en un entero de 16 bits. Te dejo un post que explica lo sucedido.

6 – Mars Climate Orbiterm (1999). 655 millones de dólares.

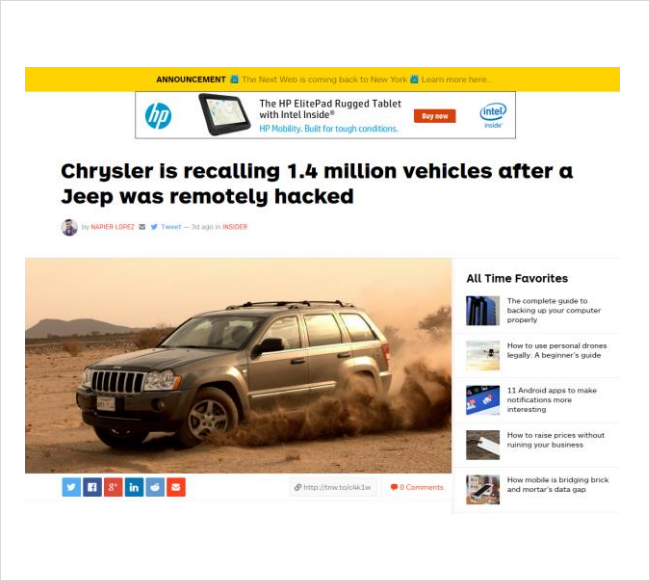
En 1999 los ingenieros de la NASA perdieron el contacto con la Mars Climate Orbiter en su intento que orbitase en Marte. La causa, un programa calculaba la distancia en unidades inglesas (pulgadas, pies y libras), mientras que otro utilizó unidades métricas.

7 – El error en los frenos de los Toyota (2010). 3 billones de dólares.

Toyota retiró más de 400.000 de sus vehículos híbridos en 2010, por un problema software, que provocaba un retraso en el sistema anti-bloqueo de frenos. Se estima entre sustituciones y demandas el error le costó a Toyota 3 billones de dólares.

8 – Las migraciones por el año 2000. 296,7 billones de dólares.

Se esperaba que el bug Y2K paralizase al mundo a la medianoche del 1 de enero 2000, ya que mucho software no había sido previsto para trabajar con el año 2000. El mundo no se acabó, pero se estima que se gastaron 296,7 billones de dólares para mitigar los daños.



Y sigue habiendo errores…

http://www.lanacion.com.ar/1663594-toyota-debera-revisar-19-millones-de-autos-prius-por-problemas-de-software

Y cuando nos va bien es por…

1. Involucramiento del usuario 15.9 %
2. Apoyo de la Gerencia 13.0 %
3. Enunciado claro de los requerimientos 9.6 %
4. Planeamiento adecuado 8.2 %
5. Expectativas realistas 7.7 %
6. Hitos intermedios 7.7 %
7. Personas involucradas competentes 7.2 %

Y cuando nos va mal es por…

1. Requerimientos incompletos 13.1 %
2. Falta de involucramiento del usuario 12.4 %
3. Falta de recursos 10.6 %
4. Expectativas poco realistas 9.3 %
5. Falta de apoyo de la Gerencia 8.7 %
6. Requerimientos cambiantes 8.1 %

Conclusión….

Saber programar NO es suficiente!!!!

Ingeniería de Software

- Parmas [1987] definió a la ingeniería en software como “multi-person construction of multi-version software”


Software Quality
Software Quality Fundamentals
Software Quality Management Processes
Practical Considerations
Software Quality Tools
Software Engineering Economics
Software Engineering Economics Fundamentals
Life Cycle Economics
Risk and Uncertainty
Economic Analysis Methods
Practical Considerations
Software Requirements
Software Requirements Fundamentals
Requirements Process
Requirements Elicitation
Requirements Analysis
Requirements Specification
Requirements Validation
Practical Considerations
Software Requirements Tools
Software Design
Software Design Fundamentals
Key Issues in Software Design
Software Structure and Architecture
User Interface Design
Software Design Quality Analysis and Evaluation
Software Design Notations
Software Design Strategies and Methods
Software Design Tools
Cuerpo de Conocimientos de la Ingeniería de Sw (SEBOK)
Software Engineering Process
Software Process Definition
Software Life Cycles
Software Process Assessment and Improvement
Software Measurement
Software Engineering Process Tools
Software Construction
Software Construction Fundamentals
Managing Construction
Practical Considerations
Construction Technologies
Software Construction Tools
Software Testing
Software Testing Fundamentals
Test Levels
Test Techniques
Test-Related Measures
Test Process
Software Testing Tools
Software Maintenance
Software Maintenance Fundamentals
Key Issues in Software Maintenance
Maintenance Process
Techniques for Maintenance
Software Maintenance Tools
Software Engineering Management
Initiation and Scope Definition
Software Project Planning
Software Project Enactment
Review and Evaluation
Closure
Software Engineering Measurement
Software Engineering Management Tools
Software Engineering Professional Practice
Professionalism
Group Dynamics and Psychology
Communication Skills
Software Engineering Models and Methods
Modeling
Types of Models
Analysis of Models
Software Engineering Methods
Software Configuration Management
Management of the SCM Process
Software Configuration Identification
Software Configuration Control
Software Configuration Status Accounting
Software Configuration Auditing
Software Release Management and Delivery
Software Configuration Management Tools

SWEBOK®

- Cuerpo de Conocimiento de la Ingeniería de Software
- Versión 3.0 del 2014 de la IEEE
- Está conformado por 15 áreas de conocimiento

Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

Ingeniería de Software: la materia en contexto

Disciplinas Técnicas

- Requerimientos
- Análisis y Diseño
- Construcción
- Prueba
- Despliegue

Disciplinas de Gestión

- Planificación de Proyecto
- Monitoreo y Control de Proyectos

Disciplinas de Soporte

- Gestión de Configuración de Software
- Aseguramiento de Calidad
- Métricas

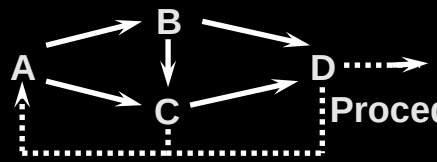
El proceso de Software

- Conjunto estructurado de actividades para desarrollar un sistema de software
- Estas actividades varían dependiendo de la organización y el tipo de sistema que debe desarrollarse.
- Debe ser explícitamente modelado si va a ser administrado.

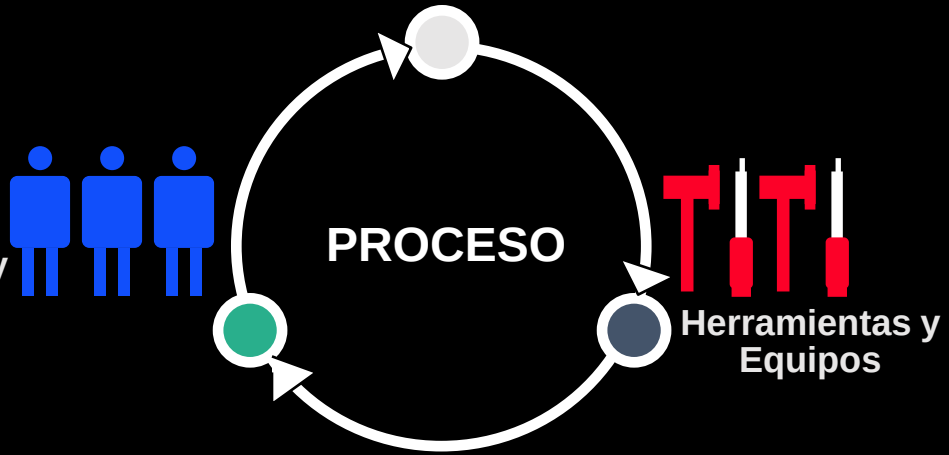
Definición de un Proceso de Software

Proceso: La secuencia de pasos ejecutados para un propósito dado (IEEE)

Proceso de Software: Un conjunto de actividades, métodos, prácticas, y transformaciones que la gente usa para desarrollar o mantener software y sus productos asociados (Sw-CMM)



Procedimientos y métodos



Personas con habilidades, entrenamiento y motivación

Herramientas y Equipos

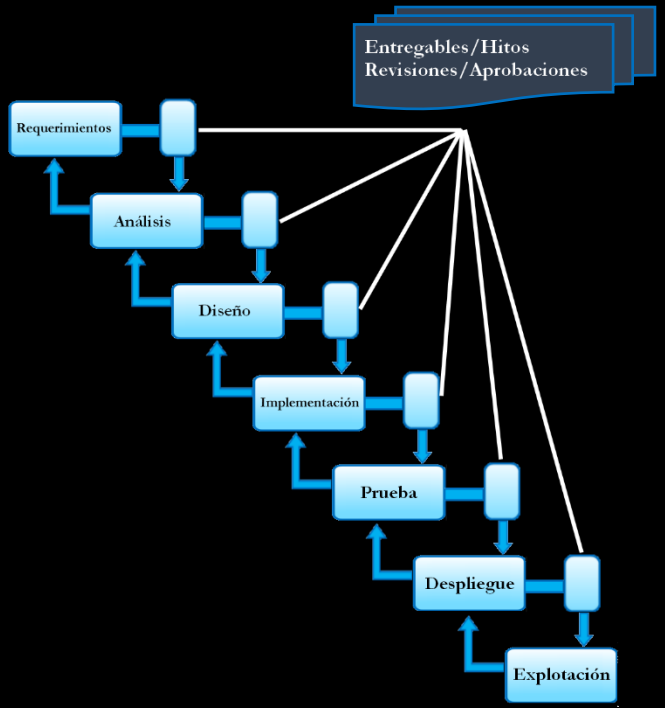
Definido vs. Empírico

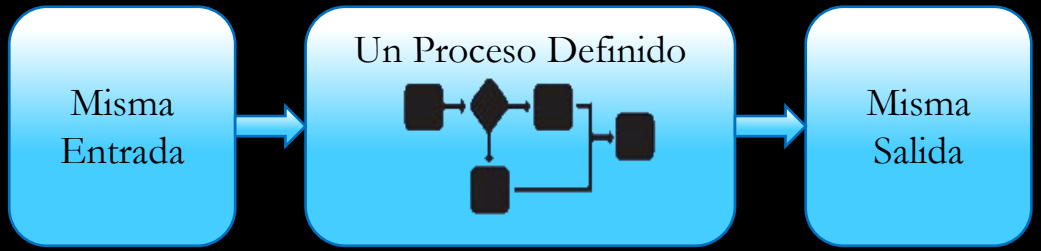
En la Universidad de California, en Irvine, simplemente sembraron pasto y esperaron 1 año, luego de eso se fijaron donde la gente había hecho “caminito”, entonces ahí construyeron las sendas peatonales

Definido (inspirados en las líneas de producción)

- Asume que podemos repetir el mismo proceso una y otra vez, indefinidamente, y obtener los mismos resultados.
- La administración y control provienen de la predictibilidad del proceso definido.

Procesos Definidos



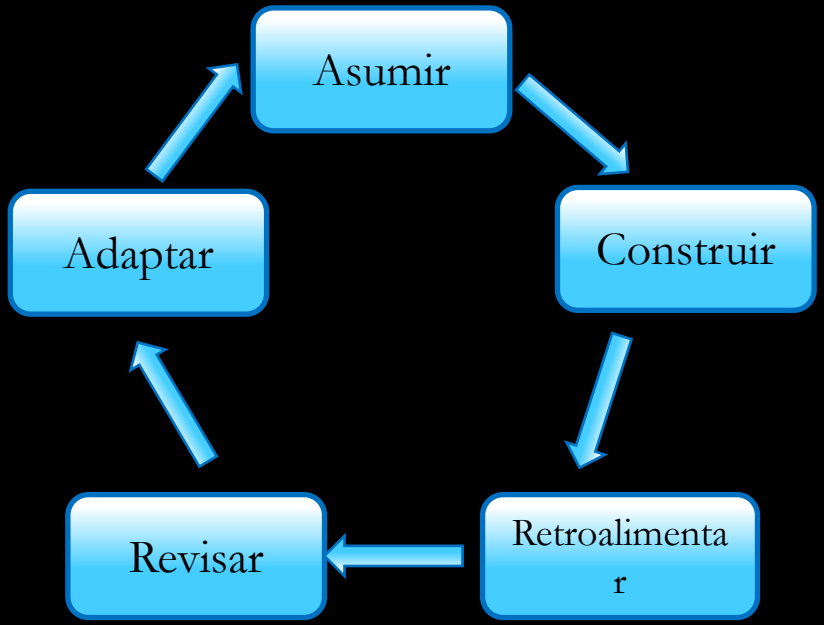


Procesos Empíricos

Empírico

- Asume procesos complicados con variables cambiantes. Cuando se repite el proceso, se pueden llegar a obtener resultados diferentes.
- La administración y control es a través de inspecciones frecuentes y adaptaciones
- Son procesos que trabajan bien con procesos creativos y complejos. (a que suena??)

Patrón de conocimiento en procesos empíricos



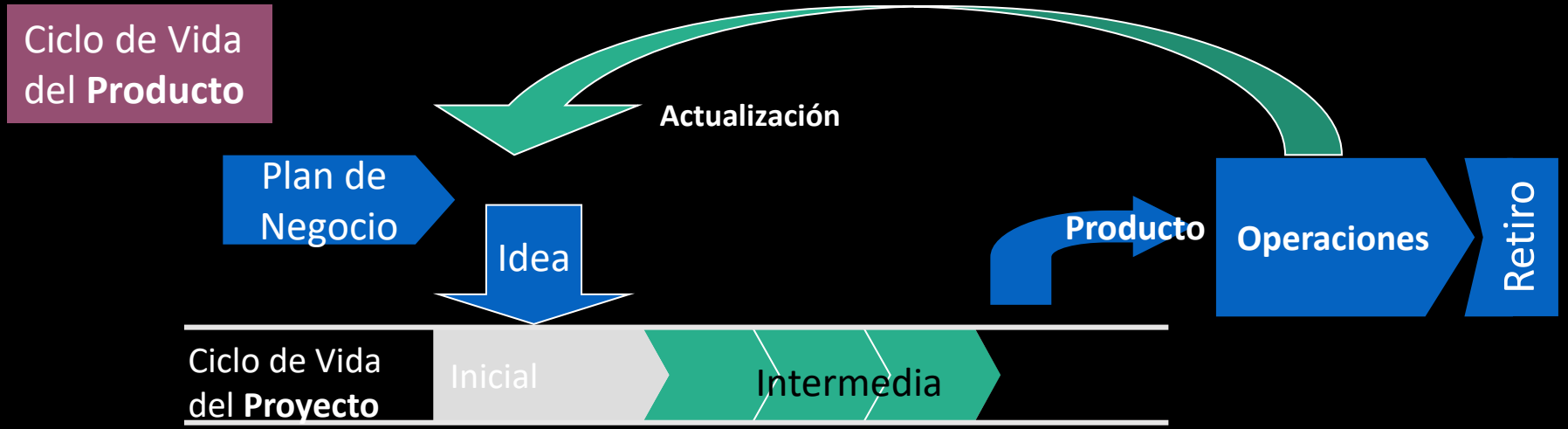
Ciclos de vida

La serie de pasos a través de los cuales el producto o proyecto progresa.

Los productos tienen su ciclo de vida.

Los proyectos también.

Relación: Ciclo de Vida del Proyecto y del Producto



Ciclos de Vida de proyectos de software

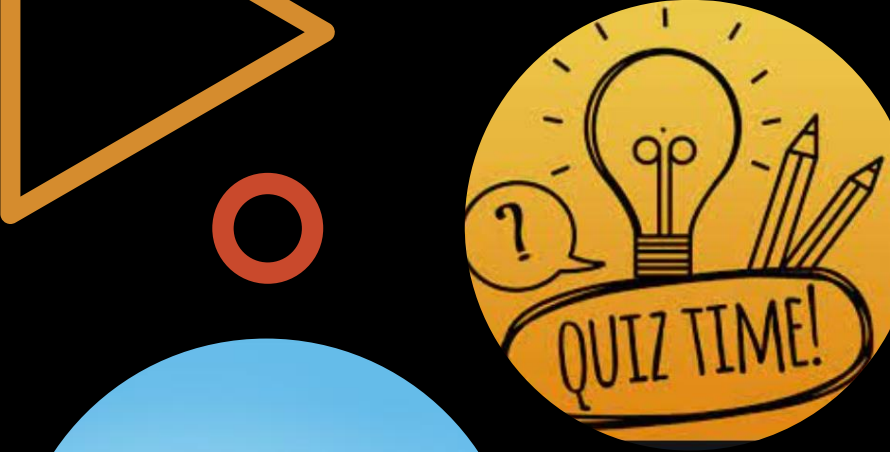
- Un ciclo de vida de un proyecto software es un representación de un proceso. Grafica una descripción del proceso desde una perspectiva particular
- Los modelos especifican
  - Las fases de proceso.
    - Ejemplo: requerimientos, especificación, diseño…
  - El orden en el cual se llevan a cabo

Clasificación de los ciclos de vida

- Hay tres tipos básicos de Ciclos de Vida
  - Secuencial
  - Iterativo/Incremental
  - Recursivo

¿Qué ciclos de vida conocen?

¿Qué Relación hay entre procesos de Desarrollo y ciclos de vida?



De ciclos de vida hay más?

Si! Capítulo 7 de Desarrollos de proyectos informáticos (Rapid Development) de Mcconell

Referencias Bibliográficas Principales

- Libros:
  - Sommerville, Ian - INGENIERÍA DE SOFTWARE - Novena Edición (Editorial Addison-Wesley Año 2011). -
  - Brooks, Frederick The mythical man-month (anniversary ed.), 1995 Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA ©1995
  - Steve Mc Connell - Rapid Development Redmond, Wa.: Microsoft Press,
  - Cohn, Mike – Agile Estimation and Planning – Editorial Prentice Hall 2006
  - Cohn, Mike - User Stories Applied – Editorial Addison Wesley 2004.
- Papers:
  - Orphans Preferred (http://www.stevemcconnell.com/psd/07-orphanspreferred.htm)
  - No Silver Bullet (http://www.virtualschool.edu/mon/SoftwareEngineering/BrooksNoSilverBullet.html)
  - Software's Ten Essentials (http://www.stevemcconnell.com/ieeesoftware/10Essentials.pdf)
  - http://www.scrumguides.org/download.html
  - Dean Leffingwell and Pete Behrens – A user story primer (2009)
  - Manifiesto Ágil http://agilemanifesto.org/iso/es/