

**Metodologías de**

**Desarrollo de Software**

Javier Barboza Solís  Diego Barillas Valverde  Jose Cerdas González

Miguel Coto García  Oscar Santamaría Venegas  Elizabeth Villalobos Molina



Universidad Cenfotec

Ingeniería del Software

Proyecto de Ingeniería del Software I

**“Metodologías de Desarrollo de Software”**

Profesores:

Álvaro Cordero

Pablo Monestel

Equipo WAPI

San José, II Cuatrimestre, 2014

Tabla de Contenidos

1. Metodología tradicional de desarrollo Espiral 4

1.1 Historia 4

1.2 Características 4

1.3 Ciclo de vida de un proyecto desarrollado en Espiral 5

1.3.1 Modelo Espiral de seis regiones 6

1.3.2 Modelo Espiral Win Win 7

1.4 Ventajas 8

1.5 Desventajas 9

1.6 Empresas que lo han adoptado 9

2. Metodología Ágil Scrum 10

2.1 Historia 10

2.2 Características 11

2.2.1 Roles Scrum 11

2.2.2 Reuniones en Scrum 13

2.2.3 Sprint 15

2.2.4 Product backlog 15

2.2.5 Sprint Backlog 16

2.3 Ciclo de vida de un proyecto desarrollado en Scrum 17

2.4 Ventajas 17

2.5 Desventajas 19

2.6 Empresas que lo han adoptado 19

3. Referencias 20

Metodologías de Desarrollo de Software

# Metodología tradicional de desarrollo Espiral

## Historia

El modelo espiral fue ideado y promulgado por Barry Boehm en 1988 y se llamó modelo evolutivo espiral. Es un modelo en el que se describen las etapas el ciclo de vida del software en donde cada iteración representa una serie de actividades.

Si bien es cierto no fue el primer modelo en donde se describen iteraciones, si fue el primero en analizar su relación con el factor de riesgo existente a la hora de desarrollar software.

Originalmente Bohem propuso un modelo de cuatro regiones cuyas variantes incluyen el modelo adaptado de seis regiones y al modelo espiral win win.

## Características

La metodología Espiral es un proceso que se utiliza especialmente en la ingeniería de software la cual se fomenta en la iteración en el proceso de desarrollo, combinando actividades de desarrollo con la gestión de riesgo con el fin de minimizar el riesgo durante el desarrollo de un proyecto de software.

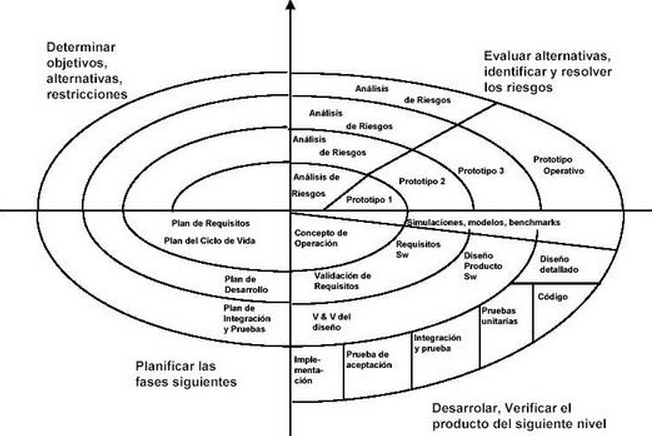
Este método vela por listar una serie de alternativas o soluciones ante cualquier riesgo que se pueda presentar durante el desarrollo del proyecto, por lo que es muy fiable para las personas que lo usan, ya que estas tienen la capacidad de optar por el riesgo más asumible y posteriormente continuar con el ciclo de la espiral.

Todo software desarrollado bajo esta metodología se desarrolla con una serie de versiones incrementales, donde al inicio se propone un prototipo sencillo y en las etapas finales se producen versiones cada vez más complejas, sin embargo puede presentarse el caso de que el cliente solicite una mejora o cambio del sistema por lo que se deben realizar todas las vueltas de espiral necesarias hasta que el producto sea aceptado y no se necesiten más cambios.

En rasgos generales los principales aspectos a tomar en cuenta por cada vuelta son:

* Objetivos: es necesario tener claro cuál es la principal necesidad que debe solucionar el producto.
* Alternativas: se deben especificar las posibles soluciones que cumplan con los objetivos.
* Desarrollar y Verificar: el desarrollo del proyecto debe ser de calidad y se tiene que asegurar que cumpla con todas las funcionalidades necesarias, de lo contrario se deberá iniciar un nuevo ciclo.

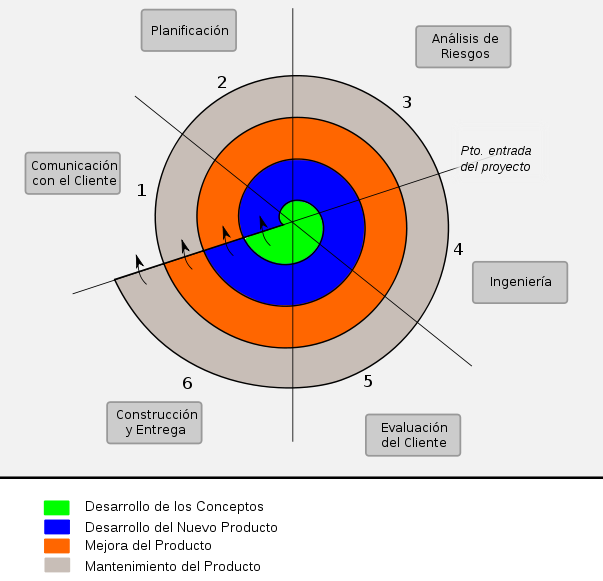
## Ciclo de vida de un proyecto desarrollado en Espiral



### Modelo Espiral de seis regiones

El modelo espiral clásico termina cuando se entrega el software sin embargo, este modelo puede adaptarse y aplicarse a lo largo de la vida del software. Las seis regiones que lo componen son las siguientes:

* Comunicación con el cliente: Se establece lo necesario para que exista un canal de comunicación apropiado entre el desarrollador y el cliente.
* Planificación: Se definen todos los requerimientos.
* Análisis de riesgos: Se determinan las acciones concretas para evaluar riesgos relacionados con el proyecto.
* Ingeniería: Se detallan las tareas necesarias para construir representaciones de la aplicación.
* Construcción y adaptación: Se realizan las pruebas e instalaciones necesarias para brindar soporte al usuario.
* Evaluación del cliente: Se definen las tareas requeridas para obtener la reacción del cliente conforme a las representaciones del software creadas.



### Modelo Espiral Win Win

Es otra variante del modelo espiral en donde se hace un énfasis en la participación del cliente en el proceso de desarrollo de un producto de software. Se establecen negociaciones para que el producto satisfaga ampliamente las necesidades del cliente y a su vez el desarrollador alcance el presupuesto y fechas de entrega razonables.



## Ventajas

1. El análisis del riesgo se hace de forma explícita y clara, por lo tanto se reducen riesgos del proyecto.
2. Al evaluarse riesgos en cada ciclo, es posible que se prevengan los problemas.
3. Al ser cíclico permite aplicar el enfoque de construcción de prototipos en cualquier etapa de evolución del producto.
4. Por su constante evaluación permite que los requerimientos no sean estáticos y puedan cambiar a lo largo del desarrollo.
5. No requiere una definición completa de los requerimientos del software a desarrollar para comenzar su funcionalidad.
6. Especial para aplicar a proyectos de gran escala por su flexibilidad y mitigación de riesgos.

## Desventajas

1. Genera mucho tiempo en el desarrollo del sistema ya que se puede llegar a iterar varias veces.
2. Requiere experiencia en la identificación de riesgos, lo cual necesita de especialistas altamente calificados.
3. Debido a su elevada complejidad no se aconseja utilizarlo en pequeños sistemas.
4. Económicamente es bastante más costoso que otras metodologías de desarrollo.
5. Por la necesidad de que el proyecto sea continuamente evaluado, es necesario que el cliente esté en el proceso de manera continua.
6. Resulta difícil convencer a clientes potenciales de que es posible controlar el enfoque evolutivo de una metodología que posee fases de desarrollo iterativas y dinámicas.

## Empresas que lo han adoptado

* Cognifide: empresa dedicada al mercadeo tecnológico. Uso de metodología espiral para el desarrollo conjunto de software (Skype, Roche, Allianz).
* SOLIDFIRE: empresa dedicada a los sistemas de almacenamiento de alto rendimiento con calidad de servicio garantizada (QoS). Uso de metodología espiral para el desarrollo de pruebas de rendimiento y seguridad.
* Healthland: empresa con énfasis en proveer servicios de información de la salud. Uso de metodología espiral para el sistema de información.
* Universidad de Sidney: institución dedicada a la educación. Uso de metodología en espiral en el área de administración de sistemas.
* eXo Platform: empresa dedicada al desarrollo de software de colaboración social entre empresas. Uso de metodología espiral para el desarrollo de software colaborativo.
* HighWire (Universidad de Stanford): división de la Universidad de Stanford dedicada al catálogo de librerías, publicaciones y referencias en línea. Uso de metodología espiral para el control de sistemas de información.
* j2 Global Communications: empresa dedicada a servicios de comunicación y mensajería. Uso de metodología espiral para las pruebas de riesgo.

# Metodología Ágil Scrum

## Historia

El concepto de Scrum tiene su origen en un estudio de 1986 (The New New Product Developement Game, por Hirotaka Takeuchi e Ikujiro Nonaka) sobre los nuevos procesos de desarrollo utilizados en productos exitosos en Japón y los Estados Unidos (cámaras de fotos de Canon, fotocopiadoras de Xerox, automóviles de Honda, ordenadores de HP y otros). Los equipos que desarrollaron estos productos partían de requisitos muy generales, así como novedosos, y debían salir al mercado en mucho menos tiempo del que se tardó en lanzar productos anteriores. Estos equipos seguían patrones de ejecución de proyectos muy similares. En este estudio se comparaba la forma de trabajo de estos equipos altamente productivos y multidisciplinares con la colaboración entre los jugadores de Rugby y su formación de Scrum (melé en español). Melé es un tipo de jugada del rugby donde todos los jugadores de ambos equipos se agrupan en una formación en la cual lucharán por obtener el balón que se introduce por el centro.

La complejidad de una melé hace que si un miembro del equipo se viene abajo, se cae toda la melé, en consecuencia, los jugadores deben estar bien coordinados, apoyarse en sus compañeros para empujar al mismo tiempo y con ello, avanzar a la misma velocidad, para aclarar de qué se trata el término.

John Scumniotales y Jeff McKenna concibieron, ejecutaron y documentaron el primer Scrum para desarrollo ágil de software en 1993, utilizando el estudio de gestión de equipos de Takeuchi y Nonaka como base.

En 1995 Ken Schwaber formalizó el proceso para la industria de desarrollo de software, de ahí expandiéndose a otras áreas y mejorándose hasta la actualidad.

## Características

Scrum es una metodología ágil y flexible para gestionar el desarrollo de software, cuyo principal objetivo es maximizar el retorno de la inversión para su empresa. Se basa en construir primero la funcionalidad de mayor valor para el cliente y en los principios de inspección continua, adaptación, auto-gestión e innovación.

### Roles Scrum

#### Scrum Master o facilitador

El *Scrum* es facilitado por un *Scrum Master*, cuyo trabajo primario es eliminar los obstáculos que impiden que el equipo alcance el objetivo del sprint. El *Scrum Master* no es el líder del equipo (porque ellos se auto-organizan), sino que actúa como una protección entre el equipo y cualquier influencia que le distraiga. El Scrum Master se asegura de que el proceso Scrum se utiliza como es debido. El Scrum Master es el que hace que las reglas se cumplan.

Durante cada *sprint*, un periodo entre una y cuatro semanas (la magnitud es definida por el equipo), el equipo crea un incremento de software *potencialmente entregable* (utilizable). El conjunto de características que forma parte de cada sprint viene del *Product Backlog*, que es un conjunto de requisitos de alto nivel priorizados que definen el trabajo a realizar. Los elementos del *Product Backlog* que forman parte del sprint se determinan durante la reunión de *Sprint Planning*. Durante esta reunión, el *Product Owner* identifica los elementos del *Product Backlog* que quiere ver completados y los hace del conocimiento del equipo. Entonces, el equipo determina la cantidad de ese trabajo que puede comprometerse a completar durante el siguiente sprint.Durante el sprint, nadie puede cambiar el Sprint Backlog, lo que significa que los requisitos están congelados durante el sprint.

#### Product Owner

El *Product Owner* representa la voz del cliente. Se asegura de que el equipo Scrum trabaje de forma adecuada desde la perspectiva del negocio. El Product Owner escribe historias de usuario, las prioriza, y las coloca en el Product Backlog.

#### Equipo de desarrollo

El equipo tiene la responsabilidad de entregar el producto. Un pequeño equipo de 3 a 9 personas con las habilidades transversales necesarias para realizar el trabajo (análisis, diseño, desarrollo, pruebas, documentación, etc).

#### Stakeholders (Clientes, Proveedores, Vendedores, etc)

Se refiere a las personas que hacen posible el proyecto y para quienes el proyecto producirá el beneficio acordado que justifica su producción. Sólo participan directamente durante las revisiones del sprint.

#### Administradores (Managers)

Son las personas que establecen el ambiente para el desarrollo del producto.

### Reuniones en Scrum

#### Daily Scrum o Stand-up meeting

Cada día de un sprint, se realiza la reunión sobre el estado de un proyecto. Esto se llama *daily standup* o *Stand-up meeting*. El scrum tiene unas guías específicas:

* La reunión comienza puntualmente a su hora.
* Todos son bienvenidos, pero sólo los involucrados en el proyecto pueden hablar.
* La reunión tiene una duración fija de 15 minutos, de forma independiente del tamaño del equipo.
* La reunión debe ocurrir en la misma ubicación y a la misma hora todos los días.

Durante la reunión, cada miembro del equipo contesta a tres preguntas:

* ¿Qué has hecho desde ayer?
* ¿Qué es lo que harás hasta la reunión de mañana?
* ¿Has tenido algún problema que te haya impedido alcanzar tu objetivo? (Es el papel del ScrumMaster recordar estos impedimentos).

#### Scrum de Scrum

Cada día normalmente después del “Daily Scrum”:

* Estas reuniones permiten a los grupos de equipos discutir su trabajo, enfocándose especialmente en áreas de solapamiento e integración.
* Asiste una persona asignada por cada equipo.

La agenda será la misma que la del Daily Scrum, añadiendo además las siguientes cuatro preguntas:

* ¿Qué ha hecho tu equipo desde nuestra última reunión?
* ¿Qué hará tu equipo antes que nos volvamos a reunir?
* ¿Hay algo que demora o estorba a tu equipo?
* ¿Estás a punto de poner algo en el camino del otro equipo?

#### Reunión de Planificación del Sprint (Sprint Planning Meeting)

Al inicio del ciclo Sprint (cada 15 o 30 días), una “Reunión de Planificación del Sprint” se lleva a cabo.

* Seleccionar qué trabajo se hará
* Preparar, con el equipo completo, el Sprint Backlog que detalla el tiempo que tomará hacer el trabajo.
* Identificar y comunicar cuánto del trabajo es probable que se realice durante el actual Sprint.
* Ocho horas como límite.
* Al final del ciclo Sprint, dos reuniones se llevarán a cabo: la “Reunión de Revisión del Sprint” y la “Retrospectiva del Sprint”.

#### Reunión de Revisión del Sprint (Sprint Review Meeting)

* Revisar el trabajo que fue completado y no completado
* Presentar el trabajo completado a los interesados (alias “demo”)
* El trabajo incompleto no puede ser demostrado
* Cuatro horas como límite

#### Retrospectiva del Sprint (Sprint Retrospective)

Después de cada sprint, se lleva a cabo una retrospectiva del sprint, en la cual todos los miembros del equipo dejan sus impresiones sobre el sprint recién superado. El propósito de la retrospectiva es realizar una mejora continua del proceso. Esta reunión tiene un tiempo fijo de cuatro horas.

### Sprint

El Sprint es el período en el cual se lleva a cabo el trabajo en sí. Es recomendado que la duración de los sprints sea constante y definida por el equipo con base en su propia experiencia. Se puede comenzar con una duración de sprint en particular (2 o 3 semanas) e ir ajustándolo con base en el ritmo del equipo, aunque sin relajarlo demasiado. Al final de cada sprint, el equipo deberá presentar los avances logrados, y el resultado obtenido es un producto potencialmente entregable al cliente. Asimismo, se recomienda no agregar objetivos al sprint o *sprint backlog* a menos que la falta de estos objetivos amenace al éxito del proyecto. La constancia permite la concentración y mejora la productividad del equipo de trabajo.

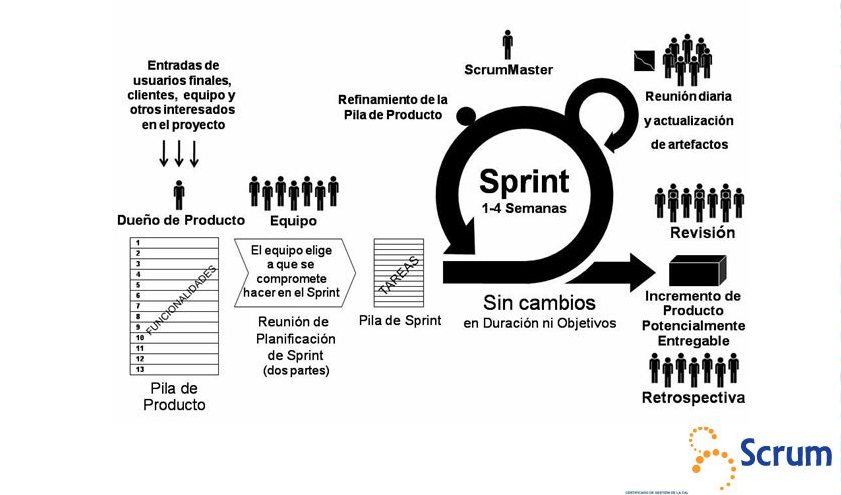
### Product backlog

El *product backlog* es un documento de alto nivel para todo el proyecto. Contiene descripciones genéricas de todos los requisitos, funcionalidades deseables, etc. priorizadas según su retorno sobre la inversión. Es el *que* va a ser construido. Es abierto y solo puede ser modificado por el *product owner*. Contiene estimaciones realizadas a grandes rasgos, tanto del valor para el negocio, como del esfuerzo de desarrollo requerido. Esta estimación ayuda al *product owner* a ajustar la línea temporal (KEV) y, de manera limitada, la prioridad de las diferentes tareas. Por ejemplo, si dos características tienen el mismo valor de negocio la que requiere menor tiempo de desarrollo tendrá probablemente más prioridad, debido a que su ROI será más alto.

### Sprint Backlog

La segunda etapa pasa por la definición del Sprint Backlog, documento que recoge las tareas a realizar y quién las desempeña. Es interesante asignar las horas de trabajo que va a suponer realizar cada una de ellas y asignarlas un coste. Si su volumen es muy grande, crear metas intermedias será un acierto.

## Ciclo de vida de un proyecto desarrollado en Scrum



* Toma de requisitos al cliente para cada requisito principal se crea un bloque de trabajo, llamado historia.
* EI cliente ordena los bloques de trabajo en una pila de producto según su prioridad de entrega.
* EI equipo de trabajo toma un grupo de historias, con el que trabajan durante una iteración o sprint.
* Una vez finalizado un sprint entregan al cliente el resultado del trabajo. Se vuelve al punto 2° hasta terminar la pila de producto.

## Ventajas

1. **Entrega de un producto funcional** al finalizar cada iteración, ya sea mensual, quincenal o semanal.
2. **Gestión regular** de las expectativas del cliente y basada en resultados tangibles. El cliente establece sus expectativas indicando el valor que le aporta cada requisito del proyecto y cuando espera que esté completado, comprueba de manera regular si se van cumpliendo sus expectativas, ya desde el inicio del proyecto puede tomar decisiones informadas a partir de resultados objetivos y dirige estos resultados del proyecto, iteración a iteración, hacia su meta.
3. **Resultados anticipados (time to market).** El cliente puede empezar a utilizar los resultados más importantes del proyecto antes de que esté finalizado por completo.
4. **Flexibilidad y adaptación** respecto a las necesidades del cliente, cambios en el mercado, etc. De manera regular el cliente redirige el proyecto en función de sus nuevas prioridades, de los cambios en el mercado, de los requisitos completados que le permiten entender mejor el producto, de la velocidad real de desarrollo, etc.
5. **Mitigación sistemática de los riesgos del proyecto.** Desde la primera iteración el equipo tiene que gestionar los problemas que pueden aparecer en una entrega del proyecto. Al hacer patentes estos riesgos, es posible iniciar su mitigación de manera anticipada.
6. **Productividad y calidad.** De manera regular el equipo va mejorando y simplificando su forma de trabajar.
7. **Alineamiento entre el cliente y el equipo de desarrollo.** Los resultados y esfuerzos del proyecto se miden en forma de objetivos y requisitos entregados al negocio. Todos los participantes en el proyecto conocen cuál es el objetivo a conseguir. El producto se enriquece con las aportaciones de todos.
8. **Equipo motivado.** Las personas están más motivadas cuando pueden usar su creatividad para resolver problemas y cuando pueden decidir organizar su trabajo.

## Desventajas

1. Se requiere de un Scrum Master, experto en la metodología que monitorice su cumplimiento.
2. A menos que exista una fecha definitiva de entrega, Scrum puede causar problemas con los alcances debido a que los administradores del proyecto se verán tentados a requerir nueva funcionalidad constantemente.
3. La estimación del costo y tiempo de un proyecto se verá afectada si una tarea no se define correctamente.
4. Esta metodología requiere miembros de equipo experimentados y motivados. Si el equipo consiste de personas nuevas con poca experiencia, el proyecto podría no terminarse a tiempo.
5. Plantea un problema si el desarrollo está restringido por una fecha de entrega y un precio de entrega cerrados por contrato.
6. Presupone que el cliente está muy involucrado en el desarrollo, participa de forma activa y continua, y revisa frecuentemente el avance de la funcionalidad conforme salen a la luz los sprints. Esto sin embargo no parece producirse en la mayoría de nuestros proyectos: el cliente participa, pero no hasta el punto de dedicar tiempo y recursos para revisar pequeños avances en el desarrollo.

## Empresas que lo han adoptado

* **Google**: practicó Agiles desde 1986 y adoptó Scrum hace 5 años. El creador de Scrum, Jeff Sutherland, ha trabajado con los equipos de Google proveyendo experiencia práctica.
* **Amazon.com**: adoptó la metodología ágil desde 1999 pero fue hasta en 2004 que la adopción de Scrum se hizo oficial.
* **Backcountry.com**: Scrum fue adoptado como metodología de desarrollo desde el año 2012 aunque ya antes se utilizaba la metodología Agile.
* **Intel**: El grupo de ingeniería de desarrollo de productos decidió coordinar la adopción de Scrum, usando 50 personas como voluntarias.
* **Microsoft.com**: La famosa empresa Microsoft adoptó la metodología Scrum hace 5 años.
* **Yahoo!**: la adopción de Scrum ha ido incrementándose desde hace años y cuentan con más de 150 equipos en Estados Unidos, Europa y Asia.

# Referencias

Acerta Software. (2014). *Metodología de desarrollo en espiral*. Recuperado de http://www.acertasoftware.com/mspiral.html

Acuña, K. B. *Por qué utilizar Scrum para desarrollar aplicaciones web.* Recuperado de http://www.eumed.net/libros-gratis/2009c/584/Por%20que%20utilizar%20Scrum%20para%20desarrollar%20aplicaciones%20web.htm

Alarcos. (2014). *Ciclo de vida del Software.* Recuperado de http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema03.pdf

Ardanaz, S., Pérez, N., Soldati, L. *Modelo Espiral*. Recuperado de http://procesosoftware.wikispaces.com/Modelo+Espiral

Atlas, A. (2009-2011) *Scrum at Amazon.* Recuperado de http://theagileexecutive.com/2009/07/20/scrum-at-amazon-guest-post-by-alan-atlas

De Soussa, S. (2009-2014) *The Advantages and Disadvantages of Agile SCRUM Software Development*. Recuperado de http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-agile-scrum-software-development.html

Elwer, P. (2008) *Agile Project Development at Intel: A Scrum Odyssey*. Recuperado de http://scrumtrainingseries.com/Intel-case-study.pdf

Fariño, G. (2011). *Modelo Espiral de un proyecto de desarrollo de software.* Recuperado de http://www.ojovisual.net/galofarino/modeloespiral.pdf

Fundación Wikimedia, Inc. (2012) Desarrollo en espiral. Wikipedia. Recuperado de http://es.wikipedia.org/wiki/Desarrollo\_en\_espiral#Variaciones\_del\_Modelo\_En\_Espiral

Grupo Espiral PHP (2009, 8 de agosto). Modelo Espiral Historia. Recuperado de http://modeloespiral.blogspot.com/2009/08/historia\_01.html

Hanantek (2007-2014). El modelo en espiral WinWin. Recuperado de http://www.hanantek.com/win-win

Marchenko, A. (2008, 14 de Noviembre) *Lessons from the Yahoo!'s Scrum Adoption*. Recuperado de http://agilesoftwaredevelopment.com/blog/artem/lessons-yahoos-scrum-adoption

Pronschinske, M. (1997-2014) *Scrum at Google*. Recuperado de http://agile.dzone.com/articles/scrum-google

QMETRY. (2014). *Featured Case Studies.* Recuperado de http://www.qmetry.com/customers

Rising, L., Janoff, N.S. (2000). The Scrum Software Development Process for Small Teams Retrieved March 15, 2007.

Slideshare.(2014).Una experiencia práctica de Scrum a través del aprendizaje basado en proyectos mediado por TIC en un equipo distribuido. Recuperado de http://www.slideshare.net/sayazyi/una-experiencia-prctica-de-scrum-a-travs-del-aprendizaje-basado-en-proyectos-mediado-por-tic-en-un-equipo-distribuido

Taft, D. K. (2005, 11 de Noviembre) *Microsoft Lauds Scrum Method for Software Projects*. Recuperado de http://www.eweek.com/c/a/IT-Management/Microsoft-Lauds-Scrum-Method-for-Software-Projects

Universidad de Oriente. (2012). *Metodologías para el desarrollo de software.* Recuperado de http://wiki.monagas.udo.edu.ve/index.php/Metodolog%C3%ADas\_para\_el\_desarrollo\_de\_software

Wikipedia. (2014). *Desarrollo en espiral*. Recuperado de http://es.wikipedia.org/wiki/Desarrollo\_en\_espiral

Wikipedia. (2014). *Software.* Proceso de creación del software. Recuperado de http://es.wikipedia.org/wiki/Software#Proceso\_de\_creaci.C3.B3n\_del\_software