Aristóteles, Dialéctica Hegeliana y Evolución de la Ingeniería de Software

Sandra P Sánchez.
Escuela Politécnica Nacional
sandra.sanchez@epn.edu.ec

Resumen

El presente trabajo contiene una caracterización de los consumidores y proveedores de software, así como un análisis de sus expectativas. Para proyectar las tendencias futuras de la Ingeniería de Software se propone utilizar el método dialéctico de Hegel en conjunción con el concepto de término medio de Aristóteles. En la parte principal, se presenta una revisión cronológica de los hitos en la evolución histórica de la Ingeniería de Software, que corresponden a tres momentos: Tesis, Antítesis y Síntesis. Finalmente, se presentan nuevas tendencias en la Ingeniería de Software, conclusiones y recomendaciones.

Palabras Claves: *consumidores de software, productores de software, término medio aristotélico, dialéctica hegeliana, hitos de la evolución de la Ingeniería de Software, tesis-antítesis-síntesis en la Ingeniería de Software*

Abstract

The present work contains a characterization of the software consumers and producers, as well as an analysis of their expectations. In order to project the future tendencies of the Software Engineering, it is proposed the use of Hegel's dialect method in conjunction with Aristotle's theory of the mean. In the main section, chronological reviews of milestones in the historical evolution of Software Engineering are presented. These milestones correspond to three moments: thesis, antithesis and synthesis. Finally, new tendencies in Software Engineering are presented as well as some conclusions and recommendations.

Keywords: *software consumers, software producers, Aristotle´s theory of the mean, Hege´s dialect method, milestones in the evolution of Software Engineering, thesis-antithesis-synthesis in Software Engineering*

1. Introducción

Para entender la evolución de la Ingeniería del Software es necesario en primer lugar caracterizar a los consumidores y proveedores de software y comprender sus expectativas.

Los consumidores de software son: la sociedad, los clientes, los usuarios (internos y externos) y usuarios no humanos, tales como otros productos de software, hardware embebido, y en general el ambiente operacional donde interactúa el software, incluyendo la nube. Por ello, el software está dejando de ser considerado un producto para ser gestionado como un servicio (SaaS – Software as a Service). Las expectativas generales de los consumidores son:

- Que el software solvente sus necesidades ejecutando funciones tal como se especificaron.
- Que desempeñe dichas funciones correcta y eficientemente a lo largo del tiempo Que sea fácil de usar y aprender
- Que sea fácil de mantener
- Que sea fácil de portar a diversos entornos

Por otra parte, los productores de software agrupan a quienes desarrollan, mantienen, gestionan y comercializan productos y/o servicios de software. Incluye a terceros involucrados tales como entidades reguladoras, certificadoras de software, entre otros. Las expectativas generales de los productores son:

- Lograr conformidad con las especificaciones del producto acorde al contrato.
- Adoptar y adaptar estándares y procesos de desarrollo con éxito.
- Seleccionar lenguajes, herramientas y entornos de desarrollo adecuados.
- Utilizar estrategias de gestión de proyectos.
- Lograr un modelo de negocio sustentable. Para lo cual es necesario: fidelización de clientes a través de satisfacer sus necesidades y brindarles servicios de mantenimiento y soporte; fidelización del personal de desarrollo, mantenimiento y soporte; y alianzas estratégicas con proveedores de tecnología y aliados de negocios.

En segundo lugar, en este artículo el concepto de término medio definido por Aristóteles y la dialéctica de Hegel son usados para entender la evolución histórica de la Ingeniería de Software y proyectar sus tendencias futuras.

2. Aristóteles y el Término Medio [1]

"Ética a Nicómaco" es una obra de Aristóteles que data del siglo IV a. C. Se trata de uno de los primeros tratados sobre ética de la filosofía occidental. Está compuesto por diez libros basados en apuntes sobre sus ponencias magistrales en el Liceo.

En el Libro II, Aristóteles afirma que la virtud es el hábito por el que la persona se hace buena y realiza bien la obra que le es confiada. La virtud se define como la disposición a elegir el medio relativo a uno en acciones y emociones, mismo que está determinado por la razón, tal como lo haría una persona prudente. Es difícil ser bueno porque es difícil encontrar el medio. Así, entre el exceso y la insuficiencia, el punto medio es la virtud. Aristóteles llama contrarios a los dos extremos más distantes, y ambos se consideran defectos.

En el Libro III, presenta como ejemplo a la fortaleza o valor, como el medio entre la cobardía y la temeridad. La persona valiente actúa a pesar del temor pero no sin temor. Otros ejemplos son: la templanza, que es el término medio respecto a los placeres; y la generosidad que es el término medio entre la avaricia y el derroche. Ver Figura 1.

Figura 1. Término Medio Aristotélico
Fuente: http://pensarenalto.files.wordpress.com/2010/11/justo-medio.jpg

Así, el término medio es aquello que se ubica a igual distancia de los dos extremos o contrarios, uno marcado por el exceso y el otro por la insuficiencia. La virtud siempre se encuentra y elige libremente el término medio.

3. Hegel y la Dialéctica [3]

Hegel expuso extensamente su filosofía en sus "Lecciones sobre la Filosofía de la Historia Universal". La pretensión fundamental de Hegel fue la de introducir la razón en la historia, es decir, intentar encontrarle un sentido, una racionalidad, mediante un método afín a la realidad misma: la dialéctica.

El método dialéctico se compone de tres momentos: tesis, antítesis y síntesis.

Tesis. Es el momento afirmativo, pero toda afirmación tiene dentro de sí una contradicción.

Antítesis. Es lo contrario de la tesis, la negación de la afirmación anterior. La realidad es conflicto, lucha de contrarios, y esa contradicción es el motor de la dialéctica. Este momento es el que dinamiza la realidad, lo que la hace moverse y evolucionar.

Síntesis. Es la superación del conflicto, la negación de la negación anterior. Los dos momentos anteriores son a la vez eliminados y conservados. La síntesis se convierte inmediatamente en tesis del proceso siguiente, que a su vez dará lugar nuevamente a una síntesis, que será a su vez, la tesis del proceso siguiente, y así sucesivamente, hasta el infinito. El proceso es continuo. Ver Figura 2.

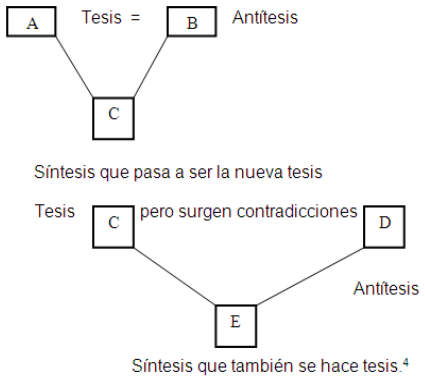

Figura 2. Momentos del Método Dialéctico
Fuente:
http://www.monografias.com/trabajos76/marxismo/image002.png

4. Evolución de la Ingeniería de Software [6] [8]

En esta sección, se presenta una revisión cronológica de los hitos en la evolución histórica de la Ingeniería de Software, que acorde al método dialéctico, corresponden a tres momentos: Tesis, Antítesis y Síntesis.

1842. El primer programa fue concebido para ser ejecutado en el "motor analítico" de Charles Babbage. Su propósito era calcular los números de Bernoulli. Lo escribió Ada Lovelace, quien es reconocida como la primera programadora.

1950. "Code and fix" es el modelo de proceso más antiguo. Consiste en producir un código, chequear si trabaja, corregir los errores y volver al paso inicial. La calidad de este código es usualmente mala y sin estructura. La corrección de errores bajo este esquema es costosa. Ver Figura 3.

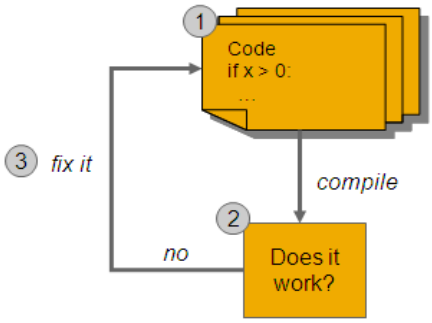

Figura 3. Code and Fix
Fuente:
http://innosophia.com/wiki/images/3/3e/Code_and_fix.png

1958. John Tukey fue el primero en utilizar el término Software en un artículo que publicó en la revista American Mathematical Monthly. Lo usó para describir los programas usados por las computadoras.

1968. Se acuña el término Ingeniería del Software en la Conferencia sobre Ingeniería de Software del Comité de Ciencia de la OTAN. Este Conferencia tenía por objetivo el tratamiento de la crisis del software provocada por el crecimiento del poder computacional del hardware.

1970. Inicia el momento dialéctico de Tesis de la Ingeniería de Software, que se caracteriza por la orientación al proceso para la obtención de software que satisfaga las necesidades del usuario. Ver Figura 4.

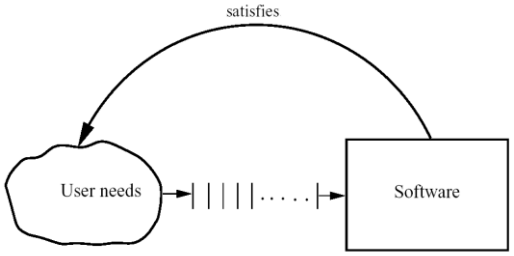

Figura 4. Orientación al Proceso de Software
Fuente: A Consice Introducction to SW Engineering [6]

1970. Winston Royce publica un artículo donde describe el desarrollo de software como un proceso secuencial, que se lo conoce como el Modelo Cascada. Ver Figura 5.

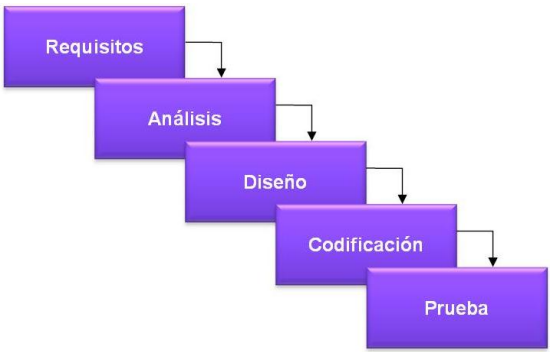

Figura 5. Modelo Cascada
Fuente:
http://blog.iedge.eu/wp-content/uploads/2011/09/IEDGE-ciclo-de-vida-desarrollo-software-2.jpg

1976. Se publica el estándar IEEE 730 Planes de Aseguramiento de la Calidad de Software. Progresivamente, la IEEE elabora una colección de estándares para Ingeniería de Software que son periódicamente revisados. La versión actual de IEEE 730 es la del año 2002.

1977. Se publica el Modelo de McCall para calidad de software, que será el origen de los estándares ISO 9216 publicado en 1991 e ISO/IEC 25000 publicado en el 2005.

1980 a 1990. Aparecen una gran cantidad de modelos de procesos de software. Entre ellos, Prototipeo, Modelo Espiral, RAD.

1991. El Instituto de Ingeniería de Software SEI publica la primera versión del Modelo de Madurez y Capacidad de Software CMM. Posteriormente se publican los procesos PSP, TSP y versiones actualizadas del modelo CMMi.

1993. Microsoft introduce el marco de trabajo MSF como un conjunto de procesos, principios y prácticas para desarrollar software.

1994. "CHAOS: A recipe for success" es publicado por Standish Group. Este reporte constituyó un punto de quiebre para la crisis del software al presentar datos estadísticos sobre los problemas en los proyectos de software. Ver Figura 6.

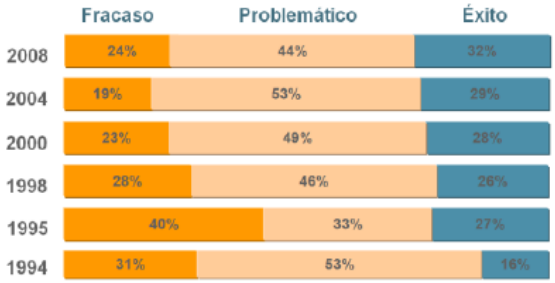

Figura 6. Reporte CHAOS
Fuente: Standish Group [8]

1996 Aparece el Proceso Unificado de Software, como una fusión de varios procesos previos y utilizando UML como notación de diseño. Posteriormente, se convierte en RUP y es adquirido por IBM en el 2003.

1996. Inicia el momento dialéctico de la Antítesis de la Ingeniería de Software, que se caracteriza por la orientación a las personas involucradas en el desarrollo del software.

1996. Sutherland y Schwaber presentan a Scrum como proceso para gestión de proyectos de desarrollo de software en la Conferencia OOPSLA. Scrum se basa en la definición de las pilas de producto y sprint, y en la realización de reuniones regulares del equipo de trabajo que es colaborativo y multifuncional. Por su versatilidad, Scrum está siendo adoptado por otras industrias. Ver Figura 7.

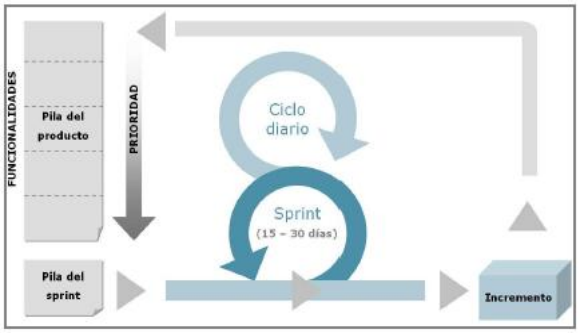

Figura 7. Proceso Scrum
Fuente: Scrum Manager [8]

1998. Se publica el Cuerpo de Conocimientos de la Ingeniería de Software SWEBOK, que caracteriza los contenidos de la Ingeniería de Software. Actualmente, está en proceso la versión 3.

1999. Beck publica el libro "Extreme Programming Explained" donde se propone un proceso que permita acortar los tiempos de desarrollo de software, a través

de enfocarse más en la programación y menos en la documentación, con un involucramiento amplio del cliente. La Programación Extrema se basa en doce principios, entre los que se destaca la programación en parejas.

2001. Varios críticos de los modelos orientados a procesos se reúnen y acuñan el término "Métodos Ágiles" para referirse a los métodos que van surgiendo como alternativa a los procesos formales. Adicionalmente, lanzan el Manifiesto Ágil que constar de cuatro postulados:

- Valorar a los individuos y su interacción, por encima de los procesos y las herramientas.
- Valorar el software que funciona, por encima de la documentación exhaustiva.
- Valorar la colaboración con el cliente, por encima de la negociación contractual.
- Valorar la respuesta al cambio, por encima del seguimiento de un plan.

Nace así una corriente humanista del desarrollo de software.

2000 a 2005 Aparecen un gran cantidad de métodos y herramientas ágiles. Entre ellos, FDD, Crystal, DSDM, Lean Software Development, Kanbam, Pomodoro.

5. Nuevas tendencias en la Ingeniería de Software [2][4] [5][7][9]

2003. Inicia el momento dialéctico de la Síntesis de la Ingeniería de Software, que se caracteriza por la búsqueda del término medio entre orientación a procesos y orientación a personas en el desarrollo de software.

2004. Boehm y Turner publican el libro "Balancing Agility and Discipline: A Guide for the Perplexed" donde establecen que estos atributos opuestos son en realidad valores complementarios en el desarrollo de software, y por ende, ambos son deseables. La clave del éxito, acorde a Boehm, es encontrar el balance correcto entre los dos.

2007. Se libera OpenUP bajo licencia libre como un proceso mínimo y suficiente para el desarrollo de software. OpenUP conserva las fases de RUP pero es de naturaleza ágil. Constituye un ejemplo de un modelo híbrido entre las dos tendencias.

2008. El Instituto de Ingeniería de Software SEI publica el reporte "CMMi or Agile: Why not embrace both!". En este reporte se explica los orígenes de los dos extremos y se establece que hay valor en ambos paradigmas. Se propone el establecimiento de una meta común de búsqueda de sinergia en pro de la evolución del desarrollo de software.

2009 a 2012. Se reportan casos de éxito de implantaciones de CMMI en MIPYMES con metodologías ágiles.

En definitiva, en este momento de Síntesis, los procesos ágiles aportan con mayor satisfacción de usuario, índices de defectos bajos, tiempos de desarrollo rápidos y apertura a los requerimientos cambiantes. Mientras que los procesos formales aportan predictibilidad, estabilidad y alto aseguramiento de calidad. Sin embargo, ambos enfoques conllevan dificultades y limitaciones. El reto está en balancear los dos extremos para tomar ventaja de sus fortalezas y compensar sus debilidades. La respuesta podría ser un enfoque basado en riesgos para estructurar proyectos que incorporen características tanto ágiles como disciplinadas en la proporción adecuada según las especificidades de cada proyecto.

6. Conclusiones

- El concepto de término medio y el método dialéctico son útiles para entender el desarrollo evolutivo de la Ingeniería de Software.
- Los métodos Ágiles y los procesos disciplinados como CMMI son elementos opuestos. Las primeras implantaciones de CMMI fueron en organizaciones muy grandes, y las primeras implantaciones ágiles fueron en empresas pequeñas, con pequeños equipos y requisitos volátiles. Ambos enfoques constituyen un par dialéctico de Tesis/Antítesis.
- Existen estudios publicados de casos reales, como el de Sutherland, uno de los creadores de Scrum, que muestran que estos dos enfoques pueden ser complementarios y potenciarse el uno al otro, logrando el punto medio que lleva a la Síntesis.
- Ambos enfoques persiguen el mismo fin. Esto es, la satisfacción de los clientes con software que cumpla sus necesidades dentro de los parámetros adecuados de costos, tiempos, y calidad.

7. Recomendaciones

- Llevar los procesos disciplinados a lo extremo, no aceptando prácticas ágiles, es tan perjudicial como llevar prácticas ágiles al extremo opuesto,

sin aceptar que la disciplina es necesaria. Se recomienda ubicarse en el punto medio.

- Se recomienda adaptar para los nuevos proyectos de desarrollo estrategias híbridas que tomen ventajas de ambos enfoques a la vez que se solventan sus debilidades, acorde a las características del proyecto.

8. Referencias

[1] Aristóteles, "Ética a Nicómaco", Introducción y Traducción de José Luis Calvo Martínez, España, 2001.

[2] Boehm M. Et al, "Observations on Balancing Discipline and Agility", Estados Unidos, 2003

[3] Gadamer H., La Dialéctica de Hegel, España, 2000

[4] Garzas J. Et al, Gestión Ágil de Proyectos de Software, España, 2012.

[5] Glazer H. Et al, CMMI or Agile: Why not embrace both!, Estados Unidos, 2008

[6] Jalote P., A Consice Introduction to Software Enginnering, Inglaterra, 2008

[7] Navarro J. Et all, Experiencia de Implantación de CMMI en una Micropyme con metodologías ágiles y software libre, España, 2010

[8] Palacio J., Scrum Manager, España, 2011 Disponible en: http://www.scrummanager.net/files/sm_proyecto.pdf (último acceso 26-julio-2012)

[9] Shuterland J. Et al, Scrum and CMMI Level 5: the magic potion for code warriors, Estados Unidos, 2007.