Nuevas formas de trabajo: La gestión del conocimiento en la industria del software.

Daniel Montes Pimentel

Eje temático 9: Estudios del trabajo

Nuevas formas de organización del trabajo

Resumen

La proliferación de industrias de bienes intangibles, como es el caso de la industria del software, permite realizar un estudio de las presumidas nuevas formas de organizar el trabajo basadas en innovación, creatividad y conocimiento. La dinámica del sector plantea un proceso conjunto entre el diseño y la ejecución, debido que parte de la subjetividad y conocimiento del trabajador. No obstante, la tendencia del sector de software es la búsqueda de la estandarización y división del trabajo para reducir costos. Lo que se observa son nuevas formas de organizar el conocimiento para lograr los objetivos en corto plazo con el menor costo. El trabajo surge del levantamiento de la información a diversas empresas de software en Querétaro.

Palabras clave: *conocimiento, formas de trabajo, fábricas de software, gestión del conocimiento, método SCRUM, ingeniería de software, aflicción del software.*

Summary

The proliferation of intangible assets industries, such as the software industry, allows a study of the presumed new forms of work organization based on innovation, creativity and knowledge. The dynamics of the sector lay dawn a joint process between design and implementation, come from subjectivity and knowledge worker. However, the trend in the software industry is seeking to standardize and divide the work to reduce costs. We are seeing new ways of organizing knowledge to achieve short-term objectives at the lowest cost. The work comes from the collection of information to various software companies in Queretaro

Key words: *knowledge, work organization, software factories, knowledge management. SCRUM model. software engineering, software affliction*

Introducción

A partir de los años 50 surgió la idea que se estaba transitando a una economía y sociedad del conocimiento. Se consideró al conocimiento como fuente de valor y creador de desarrollo económico. (Bell, 1976; Drucker, 1993; Brinkley, Fauth, Mahdon y Theodoropoulou, 2009) Por lo que es la fuente principal de innovación, el punto de partida de la productividad y la ruta para alcanzar la vía alta de desarrollo. Lo que se propone dentro esta lógica es impulsar la industria del software como un sector que cree conocimiento e implique una mayor calificación a los trabajadores consolidando su cadena

de valor y modificando la forma de producir. Los sectores intensivos en conocimiento (software, aeronáutico, nanotecnología) exigen trabajadores capacitados en nuevas tecnologías, polivalentes y con una gran capacidad de aprendizaje y adaptación a los nuevos avances. Implica un cambio estructural dentro de las industrias y nuevas configuraciones socio-técnicas volcadas a la innovación y creación de conocimiento.

El sector del software se gesta bajo esta lógica a la par del crecimiento de la producción de computadoras; pero no es hasta que se vende por separado del hardware que nace el sector como tal a nivel mundial. (Rodríguez. 2011; Shapiro, 1997) En 1960´s con la crisis del sector del software, término propuesto por Dijkstra en una conferencia para aludir a las fallas en programación y el tiempo de creación de un software, además de que el software no cumplía con los requerimientos solicitados. Se plantean los primeros aportes para sistematizar la creación de un software y así reducir la cantidad de bugs (Rodríguez, 2011; Pressman 2001; Shapiro, 1997) Surge el termino de Ingeniería de Software para designar *"el establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales*" (Pressman, 2001: 20). El Instituto de Ingenieros Eléctricos y Electrónicos (IEE, por sus siglas en Inglés) la define como: "*La aplicación de un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento de software"* (Citado en Pressman, 2001:20)

El sector del software adquiere relevancia porque genera un tipo de bien inmaterial de vital importancia para el funcionamiento de los recursos electrónicos concretos. Se ha convertido en un insumo básico para las industrias y las nuevas tecnologías. Hoy en día dentro del sector la "*calidad del software*" es el término que se ha puesto en voga y se refiere a la búsqueda de reducción de costo, tiempo y errores en la creación del software, que garantice

la satisfacción del cliente o usuario. La calidad del software como producto y como proceso de creación que tenga un valor comercial. Por tal razón "*el constante esfuerzo de una organización con la decisión de sus directivos, por hacer las cosas cada vez mejor y crear productos o servicios satisfactorios para los cliente, de una manera institucionalizada, teniendo sus miembros ciertas motivaciones, creencias, valores hábitos y conocimientos suficientes para lograr los objetivos y metas que se establezcan* "(Luna, 2009:81)

Pese a la búsqueda de la calidad del software, la *aflicción del software* sigue siendo una constante en la actualidad en el sector del software. Aún i) el sobrepaso de los costos calculados, ii) el incumplimiento de la planificación en cuestión de tiempo, iii) la falta de mejoramiento de la productividad y la eficiencia de los programadores, iv) los errores y fallos en los programas, v) falta de coordinación y bitácoras que faciliten el análisis de los errores y del mantenimiento del software, vi) falta de un procedimiento formal, sistematizado para recoger los requerimientos del cliente, vii) falta de la documentación correspondiente al software (Rodríguez, 2011) La tendencia actual "*lanzar el software antes que otro lo haga*" obliga a lanzar al mercado software con bugs que repercuten en el servicio o caídas posteriores del software. Un software es uno de los productos que puede funcionar con fallas y posteriormente puede ser reparado desde una actualización. Empero, con la creación de la nube y plataformas que almacenan diversas aplicaciones se puede hacer mantenimiento o mejoras a partir de las sugerencias de los usuarios por lo cual un software está inmerso en una mejora constante. Así mismo, la modificación de la tecnología y la creación de nuevos lenguajes de programación obligan a transitar el software a estas nuevas plataformas.

En la actualidad el sector no cuenta con una configuración sociotécnica definida, sino con métodos agiles de diseño y programación del software que permitan sociabilizar el conocimiento y reducir bugs. La tendencia de negocios "*start up*" y la gran cantidad de plataformas crea un mercado competitivo, abierto, dinámico y acoplado a lenguajes muy específicos. A la par del desarrollo de software libre y los freelance modifican el mercado del sector y obliga a las empresas a implementar una organización del trabajo para la reducción de gastos y mejorar la competitividad que les permita sobrevivir en este mercado global.

El conocimiento en el sector del software.

Dentro de la economía del conocimiento, el conocimiento se transmite y se utiliza intensivamente, *se convierte en un bien* que se produce, reproduce y se vende. Ante esta situación, la forma de organizar el trabajo en el sector del software busca acumular el conocimiento dentro de la firma para que le facilite la actualización o mantenimiento al software en años futuros, así como para crear una base de conocimientos que le permita desarrollar software más complejos a partir de proyectos previos que le faciliten la programación. Esto conlleva a tener un control excesivo y a organizar el trabajo para la transmisión de conocimiento para ser conservado por la empresa y regístralo bajo su marca.

El conocimiento tiene que codificarse, es decir, de transformarse en representaciones simbólicas que puedan ser almacenadas en un medio particular. Nonaka y Takeushi marcaban la diferencia de conocimiento tácito y el explícito. El conocimiento tácito, vital para la programación de un software, está profundamente enraizado en la acción y la

experiencia de un individuo, así como en los ideales, creencias, valores o emociones que una persona abraza (Takeushi, 2006). Por lo que la utilitarización del conocimiento encuentra rápidamente una limitante, no todo conocimiento puede ser sistematizado, todo conocimiento tiene un aspecto tácito, "resultado de la experiencia, sabiduría y creatividad de los sujetos". (Pérez y Coutin, 2005: s/n) esto representa una limitante para crear un flujo de conocimientos que permitieran una innovación constante. Si bien es cierto que nunca se le va a quitar el conocimiento al trabajador, se busca que lo sociabilice, escriba y sistematice toda actividad desarrollada para conservar el conocimiento.

Ahora bien, el conocimiento y las formas en que este se preserva y se transmite, son de vital interés para la industria del software, dado que estas definen formas concretas de organización del trabajo. Dentro de la programación del software existen dos momentos vitales: *i)* diseño del software: dónde se especifica los lineamientos y se concentra el código fuente del software y *ii)* la programación del software: si bien el programador tiene poco conocimiento del desarrollo del software, este implica en demasía la habilidad y conocimiento del programador para realizar las actividades encargadas y resolver conflictos sobre la marcha. Otra situación dentro del sector del software es que no entrega el conocimiento a los clientes o usuarios si se pacta una Licencia de desarrollo Interno donde "*Se prohíbe cualquier Modificación del Software Bajo Licencia excepto cuando sea realizado por Empresa 1 o sus desarrolladores. No está permitido hacer ningún otro uso del Software Bajo Licencia, y los siguientes usos (sin limitaciones) del Software Bajo Licencia quedan expresamente prohibidos, en la medida en que la ley aplicable permita dicha prohibición"* (Extracto de contrato entre una empresa y un cliente, 2014) lo que permite a la empresas seguir conservando el conocimiento. A diferencia de una licencia

abierta dónde todos los códigos y papeleo del software son entregados al cliente, tanto los programadores, diseñadores y empresa pierden todo derecho sobre el software creado. El programador es despojado de su conocimiento por medios legales, las cláusulas de los contratos incluyen un apartado donde se especifica que no pueden desarrollar un software parecido o venderlo por lo menos un año después de dejar la firma.

En resumen, el conocimiento dentro del sector es sumamente complejo y puede crear toda una dinámica dependiendo de la forma como se diseñe o se comercialice el software, sin embargo dos momentos son cruciales para el desarrollo del software[1] y tiene que estar organizado para garantizar su calidad. Influye la comercialización del software, si es licencia cerrada el conocimiento es conservado en la empresa. Si se trata de una licencia abierta todos tendrán el conocimiento pero la vía legal les impide utilizarlo y exige toda una documentación exhaustiva del software, lo que implica una organización del trabajo basada en documentación.

Formas de organizar el trabajo en el sector del software

El funcionamiento y desarrollo del software obedece a las lógicas económicas del conocimiento, por lo que requiere de un mercado de trabajo calificado, exige una reestructuración de los trabajadores a nuevas calificaciones y formas de *ser* dentro de las fábricas. A lo anterior se le suma una forma de configuración productiva flexible, colectiva y por proyectos. En consecuencia, las nuevas configuraciones sociotécnicas del trabajo parten de una nueva relación cliente-empresa. El procedimiento de creación dentro del

[1] I) diseño y ii) programación

sector del software parte del conocimiento del diseñador del software, donde los programadores ejecutan las acciones solicitadas a partir de su conocimiento y experiencia en la programación, por lo que no existe realmente una *"única forma de hacer las cosas"* (one best way) cada sujeto realiza las acciones específicas desde sí mismo para al final ensamblar todas las acciones y tener un software.

La diversidad de software obliga a revisar diversas formas de organizar el trabajo para la creación de un software. Se puede enmarcar: *i)* software embebido, *ii)* a la medida, *iii)* empaquetado *iv)* software sectorial *v)* software de alta o baja gama. En la actualidad la creación de aplicaciones móviles o de plataformas para albergar servicios se han vuelto las actividades más desarrolladas dentro de la entidad. A la par de encontrar diversas formas de crear un software dependiendo en que espacio se desarrolle, desde la forma de organización del trabajo de una empresa trasnacional, una gran empresa, una micro empresa hasta los freelance. Sin embargo todo software pasa una serie de pasos para su creación que son: *i)* petición del cliente o análisis de demanda de software *ii)* diseño del software y creación de la arquitectura del software *iii)* programación *iv)* testeo del software e *v)* implementación del software.

El sector del software no cuenta con una configuración sociotécnica sino diversas formas de organizar el trabajo que buscan codificar el conocimiento. La tendencia del sector es la estandarización. Existen diversas certificaciones que buscan mejorar los procesos de software. En México se cuenta con el Moprosoft, a nivel internacional el CMMi (Capability Maturity Model Integration), TSP/SPS (Team software process/ Personal software process) y la norma ISSO, busca documentar todo actividad para generar un respaldo, no obstante la interacción es la principal forma de sociabilizar el conocimiento.

La pregunta que permea dentro del sector del software es ¿cómo generar fluidez de información? ¿Cómo acumular conocimiento? Por lo que el método más utilizado dentro del sector en Querétaro es el método SCRUM donde dividen el desarrollo del software en: planeación, análisis, diseño, construcción y prueba e implementación. Se considera un desarrollo ágil debido a que en todo momento se analizan todas las acciones para resolver de forma versátil los cambios y problemas que surjan sin una atadura o documentación exhaustiva del software. Dentro de esta forma de organizar el trabajo se creó un manifiesto del desarrollo ágil que consta de cuatro puntos: *i)* A los individuos y su interacción, por encima de los procesos y las herramientas. *ii)* El software que funciona, por encima de la documentación exhaustiva. *iii)* La colaboración con el cliente, por encima de la negociación contractual, y *iv)* La respuesta al cambio, por encima del seguimiento de un plan. (Manifiesto ágil, 2014)

Estos elementos garantizan una forma dinámica de trabajo que involucra a todos los programadores, sin embargo, existe un líder, el dueño del software y el equipo, con juntas diarias se resuelven todas las problemáticas de manera inmediata y se pueden dar los cambios necesarios para cumplir con el tiempo y especificaciones demandadas por el cliente. Se logra dividiendo la programación en pequeñas “carreras” (sprint) para que el equipo se auto-organice y logre las diferentes “carreras” en los tiempos establecidos. (Manifiesto Ágil, 2014)

No obstante, al ser una empresa de software tienen una mezcla de desarrollo ágil con la ingeniería de software donde los costos, el recurso humano y el tiempo son establecidos y utilizados para medir la calidad del software. Se realiza el papeleo mínimo para cumplir la parte legal del software y la empresa tenga respaldo para registrarlo y desarrollarlo.

La organización del conocimiento en las pequeñas empresas de software:

Las pequeñas empresas de software, tienen un proceso de creación del software artesanal completamente distinto a los grandes corporativos. Todo el proceso de desarrollo del software es llevado a cabo por pocas personas y es programado por las mismas, por lo cual no se puede enmarcar que exista una diferencia entre diseño y ejecución. La creación de software es resultado de un juego de subjetividades entre el cliente y los programadores donde se mezcla la cualificación, destreza y creatividad del equipo para generar símbolos. La situación de las pequeñas empresas de software es la pérdida de conocimiento y el aumento del precio del software por demasiados bugs. La problemática que encuentran las pequeñas empresas es la transmisión del conocimiento dentro de los programadores. "*La dificultad que tenemos es que si lo programa sólo una persona y se va tenemos que hacer todo el procedimiento de nuevo*" (Empleado 1, 2014)

Al carecer de recursos económicos recurren a contratar personal no tan calificado y sin certificaciones, por lo que aprenden haciendo, usando e interactuando con las herramientas disponibles. Lo que dificulta implementar el método SCRUM a la perfección y genere atrasos en la programación del software. La mayoría de las pequeñas empresas busca empezar a implementar la cultura del "papeleo del software" ya que es parte fundamental para las actualizaciones o dar el soporte técnico del software, sin tener éxito aún por la falta de personal y tiempo para llevar a cabo a la par el papeleo necesario. Sólo cuando se da licencia abierta, se genera el papeleo mínimo para la entrega del paquete del software.

La organización del conocimiento en las grandes empresas.

Las empresas de software consolidadas en el estado, por decirlo de alguna forma ya que algunas tienes de 5 a 10 trabajadores, son aquellas que desarrollan software sectorial, software empaquetados o plataformas que les permite posicionarse en un mercado más amplio y desarrollar un modelo SCRUM, debido a que se incrementa la capacidad del software porque cuentan con documentación necesaria, la mayoría conoce el código fuente y como está en constante actualización el software los conocimientos siguen siendo transmitidos entre los programadores ya sean nuevos o con antigüedad en la empresa. Estas empresas tienen la oportunidad de desarrollar modelos de desarrollo ágil o de SCRUM en la producción de software donde el análisis, diseño, implementación y prueba están presentes en todos los momentos de creación del software. Por tal motivo, es un proceso de trabajo dinámico y cíclico que permite detectar errores o bugs en cualquier momento.

Otra vertiente son las grandes empresas nacionales que tienen desarrollado un software empaquetado y es comercializado a nivel nacional. La facilidad que tienen para capacitar a personas para instalar y dar soporte técnico permite que las empresas sólo instalen oficinas en nuevos nichos de mercado para su venta y brindando soporte técnico. Lo que hace que las oficinas instaladas desconozcan por completo el software y se conviertan en “partner comercial” sin tener la necesidad de implementar un modelo productivo o desarrollo ágil.

La organización del conocimiento en las empresas Trasnacionales.

La división del trabajo ha sido parte fundamental del desarrollo de diversos sectores industriales, el caso del sector del software no es la excepción ya que busca reducir costos y tiempo para la creación de un software. Los costos se reducen mediante la tercerialización de procesos de baja categoría, al desprenderse de la prestación de servicios de bajo valor agregado como soporte técnico, instalación y asesoría. El conocimiento se convierte en la fuente de las ventajas competitivas de las firmas, éste no fluye libremente porque está presente la censura, el poder y la lucha entre los países que dominan el área competitiva de la alta tecnología. El conocimiento se fragmenta dificultando una red de innovación integrada a países en vías de desarrollo o subdesarrollados. Por lo que, la organización del trabajo ya está segmentada y para conservar el conocimiento los líderes de proyectos son del país originario de la empresas para que no exista fuga de conocimiento (Programador 2)

Lo que ha obligado a dividir el proceso de producción del software derogando actividades que no implican desarrollo o innovación. A la par se ha segmentado la creación de un software que permite llevar a cabo diversas actividades en diferentes espacios rompiendo la característica del proceso de software diseño-programación, en ocasiones sólo se recibe la parte de soporte técnico e implementación del software en nuevos mercados sin tener una etapa de programación o diseño de software. Es decir, empresas trasnacionales pueden instalarse sin la necesidad de desarrollar software. La forma de organizar el trabajo se consolidó para mantener en un grupo reducido y en países específicos los códigos fuentes y tener en diversas locaciones la programación del software.

Fábricas de software

La fábrica de software "*es un área de desarrollo dedicada a producir componentes y procesos completos para ejecución de sistemas basados en especificaciones. Opera como una línea de ensamblado basada en los planos para armado*." (Gonzáles, 2013: 3) La tendencia de creación de fábricas de software en la entidad ha puesto en evidencia las nuevas formas de organizar el trabajo en un sector intensivo en conocimiento ya que son espacios donde se puede observar la división del trabajo.

Dentro de las fábricas de software el diseño está en otro espacio y los programadores no pueden modificar las especificaciones por ningún motivo. Los programadores no saben para quién trabajan en realidad y qué software están desarrollando, sólo conocen la programación que les toca realizar, el acceso es restringido por medio de un sistema y una contraseña que es cambiada en cada proyecto. Existen tiempos y lineamientos muy específicos para que el ensamble del software no implique problemática alguna. Se exige la documentación del software que permita saber a detalle los pasos y procedimientos que hizo el programador. Si bien no existe una "*única forma de hacer las cosas*" para programar un software, al estar tan segmentado no implica un gran problema si un programador sale de la firma, ya que sólo se pierde una pequeña parte de la programación del software, otro programador puede hacerlo diferente pero bajo los mismos lineamientos y con las características solicitadas, conservando la arquitectura del software. Es por ello, que la documentación debe de contener el problema, el algoritmo, donde se especifica el pseudocódigo y un diagrama de flujo, por último se especifica el lenguaje utilizado. (Departamento de Ingeniería de Sistemas y Automática, 2014)

Las fábricas de software no sólo realizan la programación, también pueden realizar el testeo del software, este proceso consta de probar la calidad del software analizando si cumple con las expectativas y requerimientos solicitados. El “tester” recibe por medio del sistema las partes del software que tiene que evaluar y en el mismo sistema envía lo observado, de tal manera que quede un registro y se lo envía nuevamente al programador asignado, este procedimiento se repite hasta que el programador resuelva la problemática y el tester de por visto bueno las resoluciones. Esta actividad es monótona y repetitiva, no es necesario conocer los códigos fuentes, para quién es el software o qué tipo de software es. Al ser tan segmentada la programación el testeo conserva esa segmentación y sólo verifica partes del software nunca el funcionamiento completo del mismo.

Las fábricas de software también pueden llevar el ciclo completo de creación de software, pero como se ha mencionado anteriormente existen métodos de control y formas legales para conservar el conocimiento. A la par, se necesitan especificaciones escritas y documentadas en el sistema que todo movimiento o acción es registrada, por lo que los diseñadores tienen un control excesivo y cada vez es menor su posibilidad de conservar el conocimiento.

Conclusiones y observaciones.

Al analizar el sector del software, parte de lo que se observa es la tendencia a buscar segmentar el trabajo para controlar el conocimiento y estandarizar el proceso de creación de un software. Se muestra cómo la programación de un software se convierte en manufactura compleja alejada del diseño y de los códigos fuentes. Sin embrago, tiene que quedar claras

las diferencias entre las empresas que han podido implementarlo y las pequeñas que buscan sobrevivir que plantean realizar formas de organizar el trabajo cooperativas, de crecimiento tanto personal como colectivo.

La tendencia a creer que este sector es intensivo en el uso de conocimiento y que todo proceso requiere de ponerlo en práctica, no encuentra fundamentos prácticos en la hora de analizar la forma en que se ha concebido su tercerización. Por lo que las presumidas nuevas formas de organizar el trabajo tienen rasgos del taylorismo, cómo Braverman mencionaba "*el taylorismo puede estar muerto, pero los principios que implica siguen siendo fundamentales para el diseño del trabajo en la actualidad"* (Citado en Hills, 1997:100) La división del trabajo, el control del tiempo están presentes en los programadores que realizan una pequeña parte del software en un tiempo predestinado por un supervisor, se estandariza el proceso y se le pide al programador que indique el tiempo necesario para desarrollarlo. La eficacia y eficiencia son términos presentes. En suma, la tendencia de crear fábricas de software implica un riesgo a la vía alta de desarrollo, ya que la organización del trabajo del sector del software se ha fragmentado, subcontratado y deslocalizado las actividades de poco valor agregado (López e Ibáñez, 2013)

Las pequeñas empresas están en el proceso de crear un acumulado de conocimientos que les permita desarrollar software a partir de conocimientos previos y fortalecer el conocimiento de los programadores, sin embargo, la falta de recursos económicos no les permite contratar a personal calificado, los programadores aprenden sobre la marcha. Por lo observado, la posibilidad es fomentar el desarrollo de software sectoriales o software empaquetados que permita acumular conocimiento y realmente incremente los conocimientos en la firma que permitan desarrollar mejoras en el software. Caso contario

en pequeñas empresas que la mayoría de las ocasiones desarrollan un software a la medida que es abandonado completamente hasta quedar obsoleto. Implica desarrollar software continuamente sin un incremento en las capacidades y generar papeleo de cada desarrollo que se haga, muchas de las veces pierde el código fuente porque la licencia que se otorga es abierta.

No se debe de olvidar que el sector del software crece a la par de los avances tecnológicos y de los nuevos lenguajes y requisitos de programación para los aparatos tecnológicos; por lo que "*La complejidad de software da lugar a una nueva situación más allá del alcance de los esfuerzos de ingeniería anteriores, obligando a los estudiantes de la asignatura de" ir fuera de las reglas*[2]." (Shapiro, 1997: 48) Será esta la característica en los próximos años o realmente el sector del software se segmentará creando una nueva división intelectual del trabajo basada en los principios del taylorismo. Lo observado en campo muestra que las empresas trasnacionales lo han logrado por la creación de fábricas de software donde la programación y testeo se han convertido en manufactura compleja. La terceriarización de procesos de poco valor agregado ha dividido el trabajo entre programadores, tester y diseñadores de software.

Las pequeñas empresas sólo sobreviven vendiendo servicios sin consolidar una metodología de trabajo y las grandes empresas han implementado el SCRUM donde podríamos enunciar una nueva forma de organizar el trabajo basado en la innovación y creatividad de los programadores, no obstante el trabajo está organizado para transmitir conocimientos y experiencias que faciliten la creación del software y evitar atrasos.

[2] Traducción propia de "The complexities of software give rise to a new situation beyond the scope of the previous engineering efforts, obliging students of the subject to 'go outside the rules.'"

Por último, un fenómeno poco estudiado como el freelance podría esclarecer el desarrollo ágil del software, ya que son los que promulgan, divulgan y fortalecen el manifiesto. Este tipo de trabajador independiente tiene características que complejizan su estudio, realizan una mayor variedad de asignaciones y tareas que puede ir cambiando en cada proyecto, al ser el único trabajador cumple con todas las actividades necesarias para realizar su trabajo. Cada proyecto implica crear una relación salarial y de trabajo con el cliente que puede ser compleja, puede acceder a instituciones formales para brindar servicios sin una relación estable o un contrato en sí Muchas veces no se cuenta con un espacio físico establecido para llevar a cabo el trabajo y cuenta con oficinas virtuales. El freelance en la economía del conocimiento no puede ser catalogado como trabajador u obrero, tampoco es un empresario en estricto sentido porque no maneja una empresa.

Implica un reto bastante enérgico desde localizarlos; analizar una dinámica cambiante por proyecto y deslocalizada, debido a que freelance pueden estar realizando una pequeña parte de un software para una empresa Europea o desarrollar completamente un software para una empresa trasnacional desde su casa o lugar que decida. Lo que resultaría una nueva forma de organizar el trabajo. ¿Cómo analizar el trabajo que se modifica constantemente por tecnologías y por el mercado o en cada proyecto?

Referencias

Bell Daniel, 1976, El advenimiento de la sociedad post-industrial. Un intento de prognosis social. España. Recuperado de http://tecale.org/documCurso/DANIEL_BELL-_El_advenimiento_de_la_Sociedad_Post-industrial.pdf

Brienkley Ian, Fauth Rebecca, Mahdon Michell y Theodoropoulou, 2009, the knowledge worker and knowledge work: A review of the existing definitions and measures. The Work Foundation. Uk

Departamento de Ingeniería de Sistemas y Automática, 2014, Tema 3: Introducción a la programación y Diagramas de Flujo. Escuela de Ingenieros. Universidad de Sevilla. Recuperado de http://nyquist.us.es/pepemaestre/INFOGITI/Tema%203%20DF.pdf

Drucker Peter, 1993, La sociedad post-capitalista. Norma Editores

González Damaris, 2013, Fábricas de software. Nortwhare

Hill Stephen, 1997, La fuerza cultural de los sistemas tecnológicos en: Díaz Rodrigo y Ma. Josefa Santos (coomp), Innovación tecnológica y procesos culturales. Nuevas perspectivas teóricas, México, FCE, pp 74-107

López Pablo e Ibañez Rafael, 2013, La conformación del modelo productivo español: El caso paradigmático de la industria del software. Lan Harremanak. Revista de Relaciones Laborales, (28). Pp 70-99

Luna Patricia, 2009, Las variables de la cultura de calidad en la industria del software en Querétaro, México. Tesis doctoral Facultad de Contaduría y Administración. Universidad Autónoma de Querétaro.

Manifiesto Ágil, 2014, recuperado de http://agilemanifesto.org/iso/es/ el día 21/04/2014

Pérez Yudith, y Coutín Adrián, 2005, La gestión del conocimiento: un nuevo enfoque en la gestión empresarial. Acimed

Pressman Roger, 2001, Software Engineering: A practitioners approach. Mc Graw Hill. Quinta Edición

Rodríguez José, 2011, Aprendizaje y resistencia en los trabajadores del software. PyV y Universidad de Sonora. México

Shapiro Stuart, 1997, Splitting the Difference: The HistoricalNecessity of Synthesis in Software Engineering. IEEE Annals of the History of Computing, Vol. 19, No. 1. Págs. 20-54

Takeushi Hirotaka, 2006, The New Dynamism of the Knowledge-Creating Company En Japan, Moving toward a More Advanced Knowledge Economy: Advanced Knowledge-Creating Companies Volumen 2. Compiladores Takeushi Nonaka y Shibata Tsutomu. World Bank Institute. Pág 1-10

Entrevistas:

Programador 1: Trabajador de empresa de software mexicana. Realizada en la ciudad de Querétaro el 22/11/2013

Programador 2: Ex trabajador de empresa de software. Realizada en la ciudad de Querétaro el 03/04/2014

Datos personales.

Daniel Montes Pimentel

Licenciado en Psicología Social por parte de la Universidad Autónoma de Querétaro

danielmontes.uaq@gmail.com

Actualmente: Estudiante de la Maestría en Estudios Multidisciplinarios sobre el Trabajo

Unidad multidisciplinaria sobre estudios del trabajo (UMEST)

Facultad de Psicología, Universidad Autónoma de Querétaro (UAQ)

Carretera a Chichimequillas S/N,

Terrenos Ejidales Bolaños, Querétaro, Qro.

C.P. 76140

Tel. 1921200 ext. 65431, 65508.

umest.uaq.aeropuerto@gmail.com