

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Celaya

Departamento de Sistemas y Computación

**Informe Técnico de Residencias Profesionales**

Implementación de base de conocimiento y administrador de proyectos en Massoluciones

**Presenta:**

José Armando Cárdenas Lara

No. De Control: 10031100

Ingeniería en Informática

**Asesor Externo**

Jessica Lisveth Silva Miranda

**Asesor Interno**

Ing*.* Donato VillagómezTorres

Celaya, Guanajuato. Junio del 2015

**Contenido**

[1.- Introducción 4](#_Toc426712666)

[2.- Justificación 6](#_Toc426712667)

[3.- Caracterización del área en que participo 7](#_Toc426712668)

[4.- Alcances y limitaciones 10](#_Toc426712669)

[5.- Objetivos generales y específicos 11](#_Toc426712670)

[5.1- Objetivo general 11](#_Toc426712671)

[5.2- Objetivos específicos 11](#_Toc426712672)

[6.- Problemas a resolver 12](#_Toc426712673)

[7.- Fundamento teórico 13](#_Toc426712674)

[7.1- ¿Qué es el Software? 13](#_Toc426712675)

[7.1.1- ¿Quién lo hace? 13](#_Toc426712676)

[7.1.2- ¿Por qué es importante? 13](#_Toc426712677)

[7.2- El software libre 14](#_Toc426712678)

[7.2.1- Ventajas del software libre 15](#_Toc426712679)

[7.2.2- Desventajas del software libre 17](#_Toc426712680)

[7.3- El software propietario 19](#_Toc426712681)

[7.3.1- Ventajas del software propietario 19](#_Toc426712682)

[7.3.2- Desventajas del software propietario 20](#_Toc426712683)

[7.4- Software como servicio (SaaS) 22](#_Toc426712684)

[7.4.1- Beneficios de SaaS 22](#_Toc426712685)

[7.4.2- Costes 23](#_Toc426712686)

[7.4.3- Tiempo 23](#_Toc426712687)

[7.4.4- Actualizaciones automáticas 24](#_Toc426712688)

[7.4.5- Seguridad 24](#_Toc426712689)

[7.4.6- Ecológico 24](#_Toc426712690)

[7.4.7- Desventajas de SaaS 24](#_Toc426712691)

[**7.4.8- ¿Cómo funciona el SaaS?** 25](#_Toc426712692)

[7.4.9 Software como servicio y SQL Azure como almacén en nube 25](#_Toc426712693)

[7.5- Bases de conocimiento 27](#_Toc426712694)

[7.5.1- Tipos de base de conocimiento 28](#_Toc426712695)

[7.5.2- Bases de conocimiento organizado 29](#_Toc426712696)

[7.6- Administrador de proyectos 31](#_Toc426712697)

[7.6.1- Definición de Administración de Proyectos 31](#_Toc426712698)

[7.6.2- Distinción entre una Metodología de Administración de Proyectos y una Metodología de Desarrollo de Software 31](#_Toc426712699)

[7.6.3- Fases de la administración de Proyectos 33](#_Toc426712700)

[7.6.4- Fases del desarrollo de Software 34](#_Toc426712701)

[7.6.5- Éxito y fracaso en proyectos de desarrollo de software 34](#_Toc426712702)

[7.7- SCRUM 36](#_Toc426712703)

[7.7.1- Beneficios de SCRUM 36](#_Toc426712704)

[7.7.2- Cómo Scrum proporciona estos beneficios 37](#_Toc426712705)

[7.7.3- El proceso SCRUM 39](#_Toc426712706)

[8.- Procedimiento y descripción de las actividades 42](#_Toc426712707)

[9.- Conclusiones 53](#_Toc426712708)

[Referencias bibliográficas 54](#_Toc426712709)

# 1.- Introducción

El proyecto “Implementación de base de conocimiento y administrador de proyectos en Massoluciones” fue implementado en la empresa Massoluciones, siendo su giro principal la tecnología en informática, las comunicaciones y el desarrollo de software.

Fue sugerido por la empresa debido a que no se tenía ningún tipo de base de conocimiento para almacenar la información valiosa propia de Massoluciones puesto que en cada desarrollo nuevo se hacen análisis, soluciones a problemas entre otros, de esta forma dicha información al no ser resguardada corría el riesgo de perderse, y en un momento dado que se necesitará nuevamente debería ser investigada o creada una vez más.

Ciertamente la administración de proyectos los cuales eran desarrollados bajo la metodología de mejora continua SCRUM y administrados mediante un software propio desarrollado por la empresa llamado AdminProsoft, este software no manejaba ningún tipo de metodología.

De esta forma se planteó un estudio completo de un software como servicio (SAAS) que fungiera como base de conocimiento para la empresa y administrador de proyectos bajo metodología SCRUM, este estudio debería contemplar todo aquel software que brindará un servicio como el que la empresa pretendía.

Como uno de los objetivos principales se propuso que el software ofreciera el manejo de permisos para los diferentes usuarios y que toda aquella información resguardada en dicho software perteneciera únicamente a la empresa.

El proyecto fue basado en una metodología de investigación científica, el cual se caracteriza por ser sistemático, ordenado, metódico, Racional/reflexivo, crítico.

Cuando se dio inicio al proyecto fueron planeadas varias juntas para hacer el plan de trabajo, de esta manera se llegó al acuerdo que cada día martes se llevaría a cabo una junta para presentar los avances de la primera fase.

Durante todo el ciclo del proyecto se analizaban y explicaban las ventajas y desventajas de los diferentes software estudiados.

Con la implementación de este software como base de conocimiento se mejoró el resguardo de la información valiosa de la empresa Massoluciones.

# 2.- Justificación

El presente proyecto consiste en el análisis, investigación e implementación de una base de conocimiento, que permita el almacenamiento a toda la información valiosa generada y la administración de los nuevos proyectos desarrollados en Massoluciones. Actualmente las instalaciones de los proyectos desarrollados se llevaban en bitácoras redactadas en documentos de Microsoft Office Word, pero estos documentos no eran almacenados en un lugar adecuado de tal forma que cuando se requería una instalación igual debía de rehacerse todo el proceso nuevamente, y posteriormente volver a documentar paso a paso.

Es por esta razón que surgió la necesidad de implementar una base de conocimiento propia para la empresa en donde todos los colaboradores de los departamentos pudiesen registrar sus experiencias y además se pudiera dar seguimiento administrativo a los proyectos, todo lo anterior mencionado es considerada información valiosa.

Debido a que la empresa Massoluciones está conformada por los departamentos de Software, Ventas, Redes y Soporte técnico se debía mantener un control de acceso mediante permisos a los diferentes empleados, además de esto se buscaba que toda la información que se almacenara sobre dicho software perteneciera única y exclusivamente a la empresa, estos fueron algunos de los puntos primordiales a investigar en cada software que se propuso.

El trabajo en equipo fue esencial para realizar este proyecto, fueron formados 3 equipos y cada uno de ellos debía aportar información e investigación relevante acerca del diferente software en línea que fuesen candidatos a fungir como base de conocimientos para la empresa.

# 3.- Caracteri**zación del área en que participo**

**Figura 1.** Se muestra la localización geográfica de la empresa Massoluciones.

Somos una empresa mexicana, especialista en el área de Tecnologías de Información, cuyo propósito es proveer servicios dirigidos a mejorar la competitividad de nuestros clientes, convirtiéndonos así en sus verdaderos asociados tecnológicos, siendo reconocidos por la calidad de nuestros productos y servicios como empresa líder en la región.

Como empresa integradora en tecnología, Massoluciones se consolida para ofrecer a sus clientes soluciones utilizando tecnología informática, entendiendo sus necesidades y haciéndolas propias, para así poder ofrecer las mejores alternativas para incrementar su competitividad

Convencidos de que la calidad de nuestros productos, la experiencia y el nivel de servicio de nuestra gente es lo que nos distingue, ponemos a su disposición nuestra experiencia y visión para buscar juntos el éxito de su empresa.

Contamos con 4 departamentos de especialización, siendo éstos: Software, Redes y Telecomunicaciones, Soporte Técnico y Comercialización. Cada departamento cuenta con el personal operativo independiente, lo que da fuerza y especialización a cada área.

Nuestra empresa está caracterizada por ser una organización de alta calidad cuenta con una amplia gama de proveedores internacionales que nos permiten ofrecerles la mejor solución. Nuestro alcance en cuestión a tipo de clientes que estamos enfocados va desde las microempresas, hasta altos corporativos, así como dependencias gubernamentales y gobiernos municipales.

Misión: Participar en el crecimiento de la región apoyando a las empresas en su desarrollo tecnológico con soluciones innovadoras y de vanguardia en TI, por medio de un desarrollo integral de nuestros colaboradores.

Visión: Ser el referente de soluciones de innovación de TI, consolidando el liderazgo regional de la empresa.

Ser una empresa financieramente sólida.

Ser una empresa atractiva para el desarrollo personal y profesional, que inspire a sus colaboradores a dar lo mejor de sí, a través de la mejora continua.

Valores:

*PROFESIONALISMO* Cumplimos eficientemente con nuestros compromisos, aunando la imagen y el respeto hacia nuestros clientes.

*CONFIANZA* Proporcionamos a nuestros clientes y colaboradores la certeza y tranquilidad en nuestros actos y soluciones.

*INTEGRACIÓN* Trabajamos en equipo, comprometidos con nuestra organización y respeto por los demás.

Durante este periodo de residencias, el cual se realiza en el área de sistemas de la empresa Massoluciones bajo la supervisión de la analista de proyectos del departamento de software la Lic. Jessica Lisveth Silva Miranda y debido al tipo de proyecto de área profesional se considera un desarrollo de software.

**Figura 2.** Organigrama de Massoluciones en el departamento de Software

# 4.- Alcances y limitaciones

4.1- Alcances

* El software será implementado en toda la empresa.
* Administración de usuarios.
* Manejo de sesiones de usuario.
* Administración de permisos administrativos por cada usuario.
* Capturar toda la información valiosa para la empresa a través de artículos a forma de wiki.
* Software es online no requiere instalación en el equipo de cómputo.
* Acceso por medio de usuario y contraseña.
* Creación de plantillas predeterminadas para la generación de los artículos de información valiosa.
* Administrador de proyectos para la empresa bajo una metodología SCRUM

4.2- Limitaciones

* Incompatibilidad con ciertos navegadores web
* Únicamente compatible con la metodología SCRUM

# 5.- Objetivos generales y específicos

## 5.1- Objetivo general

* Llevar a cabo la implementación de un software en línea como base de conocimiento para el resguardo de la información valiosa y que funja como administrador de proyectos de Massoluciones.

## 5.2- Objetivos específicos

* Realizar una investigación completa acerca de software candidato para fungir como base de conocimiento en la empresa.
* Realizar un consenso de ventajas y desventajas de cada software candidato.
* Realizar la elección del software.
* Desarrollar las diferentes plantillas para capturar la información valiosa de la empresa.
* Asignar los permisos pertinentes por cada usuario de la plataforma.
* Llevar a cabo el primero desarrollo de software bajo la administración del software elegido.

# 6.- Problemas a resolver

El problema que se presentaba en la empresa Massoluciones en el departamento de Software específicamente fue principalmente la ausencia de software que fungiera como base de conocimiento, el cual diera almacenamiento a toda la información valiosa que se generaba.

En cada proyecto que desarrollaba la empresa se generaba cierta información, dígase requerimientos, artículos de interés o resoluciones a problemas relacionados a la programación, instalaciones, etc. Por lo cual toda esta información no era almacenada en algún lugar seguro, puesto que corría el riesgo de extraviarse, y siendo este el caso se tendría que volver a generar el documento una vez más, esto consumía tiempo extra que no estaba contemplado y se tenía que pausar otras actividades de vital importancia en algunos casos.

En ese momento cada desarrollador generaba sus propios documentos en base a experiencias, de esta manera se tenía una pequeña base de conocimiento por cada desarrollador, a partir de aquí surge la necesidad de implantar un software como servicio en la empresa.

Respecto a la administración de proyectos no era manejada en la manera adecuada, el software que se utilizaba para esta tarea (AdminProsoft) no se basaba en ningún tipo de metodología, por lo que la empresa mostro la necesidad de un software como servicio que fungiera como base de conocimiento y como administrador de proyectos bajo una metodología de mejora continua SCRUM.

# 7.- Fundamento teórico

# 7.1- ¿Qué es el Software?

El software de computadora es el producto que construyen los programadores profesionales y al que después le dan mantenimiento durante un largo tiempo. Incluye programas que se ejecutan en una computadora de cualquier tamaño y arquitectura, contenido que se presenta a medida de que se ejecutan los programas de cómputo e información descriptiva tanto en una copia dura como en formatos virtuales que engloban virtualmente a cualesquiera medios electrónicos. (Pressman, 2010)

Probablemente la definición más formal de software es la atribuida a la IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), en su estándar 729: la suma total de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo. Bajo esta definición el concepto de software va más allá de los programas de cómputo en sus distintas formas: código fuente, binario o código ejecutable, además de su documentación.

Es decir, el software es todo lo intangible. Software es también conocido como programático o equipamiento lógico, esto es el conjunto de programas que puede ejecutar el hardware para la realización de las tareas de computación a las que se destina. Se trata del conjunto de instrucciones que permite la utilización del ordenador o computador (pc, personal computer). El software es la parte intangible de la computadora, es decir, programas, aplicaciones, etc.(Ramírez, 2006)

## 7.1.1- ¿Quién lo hace?

Los ingenieros de software elaboran y dan mantenimiento al software, y virtualmente cada persona lo emplea en el mundo industrializado, ya sea de forma directa o indirecta. (Pressman, 2010)

## 7.1.2- ¿Por qué es importante?

El software es importante porque afecta a casi todos los aspectos de nuestras vidas y ha invalidado nuestro comercio, cultura y actividades cotidianas. (Pressman, 2010)

# 7.2- El software libre

El «software libre» es una cuestión de libertad, no de precio. Para comprender este concepto, debemos pensar en la acepción de libre como en «libertad de expresión» y no como en «barra libre de cerveza».

Con software libre nos referimos a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software.

* Nos referimos especialmente a cuatro clases de libertad para los usuarios de software:Libertad 0: la libertad para ejecutar el programa sea cual sea nuestro propósito.
* Libertad 1: la libertad para estudiar el funcionamiento del programa y adaptarlo a tus necesidades —el acceso al código fuente es condición indispensable para esto.
* Libertad 2: la libertad para redistribuir copias y ayudar así a tu vecino.
* Libertad 3: la libertad para mejorar el programa y luego publicarlo para el bien de toda la comunidad —el acceso al código fuente es condición indispensable para esto.

Software libre es cualquier programa cuyos usuarios gocen de estas libertades. De modo que deberías ser libre de redistribuir copias con o sin modificaciones, de forma gratuita o cobrando por su distribución, a cualquiera y en cualquier lugar. Gozar de esta libertad significa, entre otras cosas, no tener que pedir permiso ni pagar para ello.

Asimismo, deberías ser libre para introducir modificaciones y utilizarlas de forma privada, ya sea en tu trabajo o en tu tiempo libre, sin siquiera tener que mencionar su existencia. Si decidieras publicar estos cambios, no deberías estar obligado a notificarlo de ninguna forma ni a nadie en particular.El software libre no significa que sea «no comercial». Cualquier programa libre estará disponible para su uso, desarrollo y distribución comercial. El desarrollo comercial del software libre ha dejado de ser excepcional y de hecho el software libre comercial es muy importante. (Stallman, 2004)

## 7.2.1- Ventajas del software libre

El software libre presenta una serie de ventajas sobre el software propietario por los derechos que otorga a sus usuarios. Algunas de estas ventajas puede ser más apreciadas por los usuarios particulares, otras por las empresas, y otras por las administraciones públicas.*Principales ventajas.*

1. Bajo costo de adquisición y libre uso. El software, como mercadería, por lo general no está a la venta. Lo que el usuario adquiere, a través de una erogación monetaria o sin ella, es una licencia respecto de los usos que puede dar a los programas en cuestión. El software no solo cuesta un precio de adquisición de licencia. También cuesta mantenerlo, operarlo, ajustarlo.

Es importante para el usuario el poder mantener estos costos bajo control, pues de lo contrario puede llegar a verse impedido de llevar a cabo sus metas, a fuerza de erogaciones no planificadas. El usuario que adquiere software libre lo hace sin ninguna erogación monetaria o a muy bajo costo y ofrece un conjunto de recursos muy amplios. Cualquier persona con una computadora y una conexión a Internet puede utilizar un software libre. Para la mayoría de usuarios individuales el software libre es una opción atractiva por las libertades que garantiza

Sin necesidad de verse agobiados por el precio. Sin embargo, en el caso de empresas y la

Administración Publica, el costo del software es un factor importante y a veces determinante en la elección de nuevos sistemas informáticos. Cuando se analiza el precio de una solución tecnológica se suele hablar del TCO (Total Cost of Ownership), es decir, del coste total de la propiedad que tiene una determinada solución de software. Este concepto fue inventado por el Gartner Group en 1987 como herramienta de análisis exhaustiva de los costos de una solución de mercado y se convirtió en un estándar. En análisis refleja el costo del programa, la ayuda, y el mantenimiento tecnológico de la solución. Si partimos de la base que el software libre prácticamente carece de costo de licencia y por lo tanto, esta parte del presupuesto se puede invertir para mejores fines: mejorar la adaptación de los programas y el desarrollo tecnológico del software. La gran mayoría del software libre tiene como postulado la determinación de que el usuario no es un iletrado computacional. Parte del hecho de que confía en los conocimientos y habilidad del usuario para lograr su objetivo. Esta situación se observa desde el diseño del Sistema operativo: decenas de pequeños comandos que hacen una tarea específica y se pueden conectar entre sí para realizar tareas más complejas. Este punto de partida otorga gran poder y flexibilidad para los usuarios capaces y con ánimo de aprender, pero limita al resto de las personas que solo quieren facilitarse una tarea de manera inmediata.

El software libre tiene las siguientes características:

* Todo el mundo tiene derecho de usarlo sin costo alguno.
* Todo el mundo tiene derecho a acceder a su diseño y aprender de él. Es como obtener las instrucciones para construir un carro.
* Todo el mundo tiene derecho de modificarlo: si el software tiene limitaciones o no es adecuado para una tarea, es posible adaptarlo a necesidades específicas y redistribuirlo libremente.
* No tiene un costo asociado (gratuito).
* Es de libre distribución (cualquier persona puede regalarlo, venderlo o prestarlo)

Estos derechos típicamente no están disponibles con el software propietario. Usualmente en el software propietario hay que pagar una “licencia de uso” al creador (como el pago de derechos por el uso de una patente) y se está sujeto a las condiciones del fabricante. Normalmente estas condiciones no otorgan ningún derecho al usuario final. Los derechos mencionados anteriormente tienen una serie de efectos colaterales sobre el Software:

* Tiende a ser muy eficiente (porque mucha gente lo optimiza, mejora).
* Tiende a ser muy robusto (mucha gente \*puede\* arreglarlo, no solamente el creador o la compañía que lo produce). Mucha gente tiende a contribuir, porque es en el interés de todos mejorar esta base común.
* Tiende a ser muy diverso: la gente que contribuye tiene muchas necesidades diferentes y esto hace que el software este adaptado a una cantidad más grande de problemas.

El ´éxito del software libre se debe en su mayor parte a Internet. Internet ha permitido que las personas interesadas en los varios componentes del software libre se pongan fácilmente en contacto con otras. Internet de esta manera actúa como un catalizador que acelera el desarrollo y sintetiza el conocimiento en ´áreas muy específicas. Hay diferentes motivaciones que impulsan a los contribuidores y desarrolladores a trabajar en el software libre, las más importantes son:

* El deseo de crear nuevo software.
* El deseo de crear software más robusto.
* La posibilidad de estar en control del software. Esto es importante para aplicaciones de misión crítica donde es imperante tener un control total sobre posibles problemas en cualquier punto.
* Crear aplicaciones de bajo costo.
* Reutilización del conocimiento: Esto permite que la gente reutilice el conocimiento que se ha sintetizado en el software. En vez de empezar siempre desde cero (que es el caso de la industria de software actual) siempre se puede empezar un proyecto desde un fundamento establecido. Esto es equivalente a la manera en la que la ciencia se desarrolla: no se parte de cero, se parte de los descubrimientos previos y se innova sobre el conocimiento que ya se tiene.
* La posibilidad de adaptar el software a sus necesidades.
* Aprender alguna técnica de programación.
* Reconocimiento de colegas.

## 7.2.2- Desventajas del software libre

**La curva de aprendizaje es mayor**. Si ponemos a dos señoras que nunca han tocado una computadora, probablemente tardaran lo mismo en aprender a usar software propietario por ejemplo de Microsoft, que software libre como Gnome o KDE; pero si antes los usuarios ya usaron software propietario generalmente tarda más en aprender a usar un software libre.

**El software libre no tiene garantía proveniente del autor.**

**Los contratos de software propietario no se hacen responsables por daños económicos, y de otros tipos por el uso de sus programas.** El software libre se adquiere se vende “AS IS” (tal cual) sin garantías explicitas del fabricante, sin embargo, puede haber garantías específicas para situaciones muy específicas.

**Se necesita dedicar recursos a la reparación de errores.** Sin embargo en el software propietario es imposible reparar errores, hay que esperar a que saquen a la venta otra versión.

**No existen compañías únicas que respalden toda la tecnología.**

**Las interfaces graficas de usuario (GUI) y la multimedia apenas se están estabilizando.** Aunque hay un número cada vez mayor de usuarios que aseguran que las interfaces graficas más populares en el software libre (KDE, GNOME y el manejador de ventanas WindowMaker) son ya lo suficientemente estables para el uso cotidiano y lo suficientemente amigables para los neófitos de la informática.

**La mayoría de la configuración de hardware no es intuitiva.** Se requieren conocimientos previos acerca del funcionamiento del sistema operativo y fundamentos del equipo a conectar para lograr un funcionamiento adecuado. Sin embargo la documentación referente a la configuración del hardware es tan explícita y detallada que permite al usuario neófito profundizar en el conocimiento de su hardware en muy pocas horas y una vez teniendo ese conocimiento la configuración se vuelve trivial.

**Únicamente los proyectos importantes y de trayectoria tienen buen soporte, tanto de los desarrolladores como de los usuarios.** Sin embargo existen muchos proyectos más pequeños y recientes que carecen del compromiso necesario por parte de sus usuarios o desarrolladores para que sean implementados de manera confiable. Estos proyectos importantes que tienen un excelente soporte cubren más del 90 % de las necesidades de cómputo del usuario promedio. (Culebro Juárez , Gómez Herrera, & Torres Sáchez, 2006)

# 7.3- El software propietario

Definimos como Software Propietario al conjunto de aplicaciones y programas que se enmarcan y protegen bajo los Derechos de Uso, Redistribución o Modificación, siendo entonces mantenidos bajo una reserva por parte de quienes lo han creado, publicado o publicitado, o bien sobre quienes mantienen y hacen uso de dichos derechos. (MasterMagazine, s.f.)

Este concepto es sin embargo una traducción errónea del Propietary Software, teniéndose en este caso el erróneo ejemplo de que no podría ser propiedad o dueño de nada, sino que se trata de un error Semántico, considerándose como propietario más bien a quien pueda ejercer un Derecho de Propiedad sobre el mismo.

Para poder solucionar las posibles confusiones respecto a este término es que se suele llamar entonces al Software con Propietario, como una de las formas de diferenciar a que justamente todo Software puede hacer uso de los derechos acerca de su distribución y utilización, definiéndose entonces a un "Software No Libre" que solamente puede ser distribuido, reutilizado o modificado bajo explícita expresión de quien tiene Derechos de Autor sobre el mismo.

## 7.3.1- Ventajas del software propietario

**Control de calidad.** Las compañías productoras de software propietario por lo general tienen departamentos de control de calidad que llevan a cabo muchas pruebas sobre el software que producen.

**Recursos a la investigación**. Se destina una parte importante de los recursos a la investigación sobre los usos del producto.

**Personal altamente capacitado.** Se tienen contratados algunos programadores muy capaces y con mucha experiencia. Uso común por los usuarios. El software propietario de marca conocida ha sido usado por muchas personas y es relativamente fácil encontrar a alguien que lo sepa usar.

**Software para aplicaciones muy específicas.** Existe software propietario diseñado para aplicaciones muy específicas que no existe en ningún otro lado más que con la compañía que lo produce

**Amplio campo de expansión de uso en universidades.** Los planes de estudios de la mayoría de las universidades de México tienen tradicionalmente un marcado enfoque al uso de herramientas propietarias y las compañías fabricantes ofrecen a las universidades planes educativos de descuento muy atractivos. De ahí que los recién egresados pueden comenzar su vida productiva utilizando estos productos de inmediato. No obstante, en los centros de estudio más prestigiados se observa un cambio en esta tendencia.

**Difusión de publicaciones acerca del uso y aplicación del software.** Existe gran cantidad de publicaciones, ampliamente difundidas, que documentan y facilitan el uso de las tecnologías proveídas por compañías de software propietario, aunque el número de publicaciones orientadas al software libre va en aumento.

## 7.3.2- Desventajas del software propietario

**Cursos de aprendizaje costosos.** Es difícil aprender a utilizar eficientemente el software propietario sin haber asistido a costosos cursos de capacitación.

**Secreto del código fuente.** El funcionamiento del software propietario es un secreto que guarda celosamente la compañía que lo produce. En muchos casos resulta riesgosa la utilización de un componente que es como una caja negra, cuyo funcionamiento se desconoce y cuyos resultados son impredecibles. En otros casos es imposible encontrar la causa de un resultado erróneo, producido por un componente cuyo funcionamiento se desconoce.

**Soporte técnico ineficiente.** En la mayoría de los casos el soporte técnico es insuficiente o tarda demasiado tiempo en ofrecer una respuesta satisfactoria. Ilegal o costosa la adaptación de un módulo del software a necesidades particulares.

**Es ilegal extender una pieza de software propietario para adaptarla a las necesidades particulares de un problema específico.** En caso de que sea vitalmente necesaria tal modificación, es necesario pagar una elevada suma de dinero a la compañía fabricante, para que sea esta quien lleve a cabo la modificación a su propio ritmo de trabajo y sujeto a su calendario de proyectos.

**Derecho exclusivo de innovación.** La innovación es derecho exclusivo de la compañía fabricante. Si alguien tiene una idea innovadora con respecto a una aplicación propietaria, tiene que elegir entre venderle la idea a la compañía dueña de la aplicación o escribir desde cero su propia versión de una aplicación equivalente, para una vez logrado ´esto poder aplicar su idea innovadora.

**Ilegalidad de copias sin licencia para el efecto.** Es ilegal hacer copias del software propietario sin antes haber contratado las licencias necesarias.

**Imposibilidad de compartir.** Si una dependencia de gobierno tiene funcionando exitosamente un sistema dependiente de tecnología propietaria no lo puede compartir con otras dependencias a menos que cada una de estas contrate todas las licencias necesarias.

**Quedar sin soporte técnico.** Si la compañía fabricante del software propietario se va a la banca rota el soporte técnico desaparece, la posibilidad de en un futuro tener versiones mejoradas de dicho software desaparece y la posibilidad de corregir los errores de dicho software también desaparece. Los clientes que contrataron licencias para el uso de ese software quedan completamente abandonados a su propia suerte.

**Descontinuación de una línea de software.** Si una compañía fabricante de software es comprada por otra más poderosa, es probable que esa línea de software quede descontinuada y nunca más en la vida vuelva a tener una modificación.

**Dependencia a proveedores.** En la mayoría de los casos el gobierno se hace dependiente de un solo proveedor.

**Nulificación de desarrollo tecnológico de la industria nacional.** Nulidad de desarrollo tecnológico de la industria nacional, respecto de la extranjera (las aplicaciones de consumo masivo se desarrollan en otros países). (Culebro Juárez , Gómez Herrera, & Torres Sáchez, 2006)

# 7.4- Software como servicio (SaaS)

SaaS (de las siglas en inglés de "Software as a Service") o software como servicio, es un término que se usa para describir cuando los usuarios "rentan" o usan prestado un software en línea, en lugar de comprarlo e instalarlo en sus propios equipos de hardware.

SaaS es una idea que ve a negocios enteros y sus miles de empleados usando herramientas de software como productos rentados en línea, a través de Internet. De esta forma, todo el trabajo de procesamiento y almacenamiento de archivos se realiza en servidores remotos que accedes a través de Internet, utilizando un navegador web.

El SaaS podría catalogarse mejor como un buen acuerdo comercial que como un programa de software, ya que es un servicio que facilita el programa de software y garantiza su mantenimiento para el cliente. Este servicio incluye una amplia gama de servicios de infraestructura informática: redes, almacenamiento, sistemas operativos, bases de datos, servidores web y servicios de restauración y de copia de seguridad. Además, ofrece un amplio abanico de servicios operativos: autenticación, disponibilidad, gestión de identidades, control de producción, gestión de paquetes de ayuda, control de la actividad, actualizaciones de la versión y personalización del software.

SaaS, cuando se combina con PaaS, constituye lo que se conoce como "computación en la nube"

SaaS y PaaS (plataforma como servicio, de las siglas en inglés de "Platform as a Service") describen el modelo de negocio de los usuarios que se conectan a un servidor centralizado en la nube para acceder a sus productos de software. Los usuarios pueden acceder a sus archivos y software sólo mientras están en línea, mediante su navegador de Internet y sus contraseñas. Es muy similar al paradigma de la década de los 50's y el modelo de terminales conectadas a un mainframe de los 60's, pero adaptado a que los navegadores de Internet hacen las veces de terminales.

## 7.4.1- Beneficios de SaaS

La principal ventaja del software como servicio es que reduce costos para todos los involucrados.

Los proveedores de software no tienen que gastar miles de horas de apoyo a los usuarios a través del teléfono, y en su lugar únicamente tienen que darle servicio y mantenimiento a una copia central del producto, disponible en Internet. Por otro lado, los usuarios no tienen que pagar los altos costos iniciales de compra de software y hardware relativamente poderoso, sino únicamente tendrían que pagar el alquiler para acceder al servicio, además de que el hardware puede ser mucho más sencillo y económico.

Si tuviéramos que resumir las ventajas del SaaS, lo haríamos del siguiente modo:

## 7.4.2- Costes

La inversión inicial es exactamente cero. Además, no tiene que comprar nuevos servidores, más espacio en disco ni ordenadores. SaaS utiliza los recursos que usted ya posee.

La realización de presupuestos resulta mucho más sencilla ya que sabe cuánto va a gastar. No habrá sorpresas que aumenten los costes inesperadamente, como nuevas versiones del software o programas que acaban fallando, ni tendrá que instalar nuevos antivirus.

De hecho, sus costes no sólo serán menores este año, sino durante todo el ciclo de uso del producto. En 2006, un estudio de McKinsey calculó que el TCO (coste total de la propiedad) del software en modo SaaS es un 40% inferior al del software instalado en el ordenador del cliente.

## 7.4.3- Tiempo

Despliegue rápido en modo de actualización continua

Configuración y funcionamiento más rápidos gracias a que el software está disponible a través de una simple conexión a Internet.

Accesibilidad en línea permanente desde cualquier punto

El hecho de que se acceda a él a través de Internet supone que el software en modo SaaS está disponible en todo momento y lugar, por lo que los empleados pueden trabajar desde cualquier punto con acceso a Internet sin necesidad de tener un software individual con licencia.

Rápida puesta en marcha

Innovación continúa

## 7.4.4- Actualizaciones automáticas

Ayuda con el cumplimiento de las normativas

Dado que el software está diseñado para un uso profesional, tiene en cuenta los requisitos profesionales, especialmente en el campo del cumplimiento normativo, que está integrado directamente en la interfaz de usuario. El software se actualiza constantemente en función de la legislación y las normativas vigentes.

Máxima reutilización de documentos

Dado que todos los documentos están almacenados y disponibles en una ubicación central, se reducen de un modo espectacular los problemas derivados del almacenamiento de datos duplicados o de la incapacidad para encontrar información esencial. Todos los documentos válidos pueden copiarse para su reutilización, lo que minimiza el tiempo que hasta ahora se dedicaba a "reinventar la rueda".

## 7.4.5- Seguridad

Proporciona un alto nivel de seguridad

El acceso a datos delicados se restringe mediante el uso de plataformas, certificados, tarjetas inteligentes y espacios de trabajo aislados. En muchos casos, el software SaaS proporciona un nivel de seguridad mayor del que la mayoría de las empresas desearían tener en el software residente en sus propias instalaciones.

## 7.4.6- Ecológico

Un beneficio inesperado del software SaaS es que resulta la solución de TI más ecológica gracias a que varios clientes comparten servidor y requisitos de almacenamiento de datos y con menos personal se puede realizar el mismo trabajo, lo que reduce la necesidad de adquirir hardware. Además, el sistema requiere menos energía ya que cuenta con muchos más recursos compartidos. E incluso se reduce el número de residuos generados por el hardware que ya no se puede utilizar, un aspecto muy a tener en cuenta, especialmente en la Unión Europea, donde las normativas relativas a la eliminación de residuos de equipos informáticos obsoletos se han endurecido recientemente.

## 7.4.7- Desventajas de SaaS

El riesgo de software como servicio es que los usuarios deben poner un alto nivel de confianza en los proveedores de software en línea, tanto en que el servicio no será interrumpido como en la privacidad de su información. En cierto modo, el proveedor del software tiene en sus manos todos los documentos y la capacidad productiva de sus clientes, por lo que la elección del proveedor toma especial importancia. Cuando una empresa de 600 empleados cambia hacia el modelo de software como servicio, debe elegir a su proveedor de software cuidadosamente. Normalmente se reduce dramáticamente el costo de administración al usar software como servicio, sin embargo, hay un aumento en los riesgos de interrupción del servicio, la conectividad y la seguridad en línea.(Castro, s.f.)

## 7.4.8- ¿Cómo funciona el SaaS?

Cada usuario conforma un grupo de usuarios en el ambiente compartido de varios grupos de usuarios del proveedor de SaaS. Puede permitirse cierta personalización del software. No obstante, no resulta habitual que el usuario tenga mucho control sobre la arquitectura de cómputo de la infraestructura, la ubicación o los niveles de servicio proporcionados.(EMC2, s.f.)

## 7.4.9 Software como servicio y SQL Azure como almacén en nube

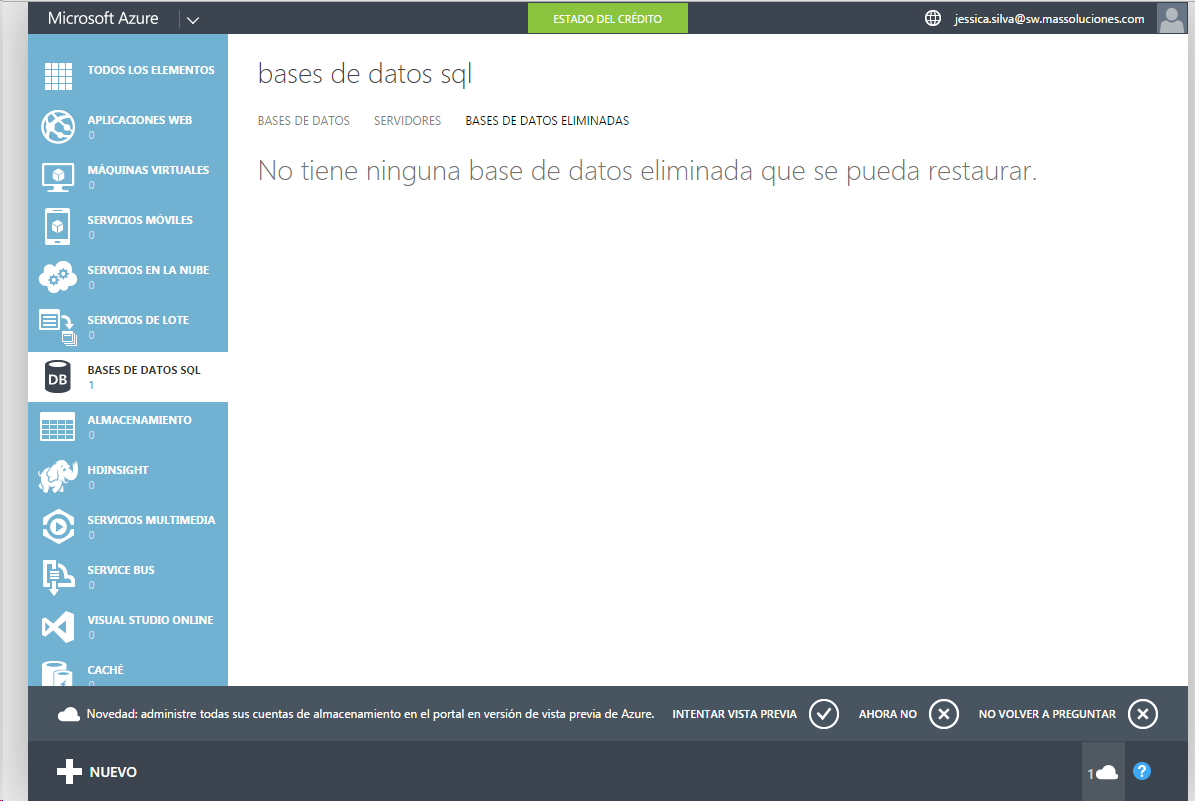
Hoy en día la mayoría de los SaaS almacenan sus bases de datos principales en la nube, garantizando así el acceso a su información en todo momento.

SQL Azure es una base de datos relacional en la nube construida sobre la tecnología de SQL Server. Proporciona servicios de bases de datos altamente escalables y con altísima disponibilidad alojados por Microsoft en la nube. Estos servicios facilitan enormemente el despliegue de bases de datos. Una ventaja añadida es que los desarrolladores y el personal de IT no necesitan instalar, actualizar y gestionar la infraestructura de bases de datos. La alta disponibilidad, aspecto siempre complejo, es gestionado de manera transparente. La gran ventaja de utilizar SQL Azure frente a otros sistemas de almacenamiento en la nube es que todos los conocimientos sobre bases de datos relacionales y el lenguaje de consulta SQL siguen siendo válidos. No es necesario adaptar los conocimientos a nuevos paradigmas de almacenamiento, como pasa con otros sistemas de almacenamiento en la nube no basados en bases de datos relacionales ni SQL. “Si sabes utilizar SQL Server, todos tus conocimientos te valen para SQL Azure”. (Ver Figura 3) (Nube, s.f.)

SQL Azure permite incluso migrar backends de datos a la nube si tener que tocar ni una sola línea de código de las aplicaciones en un gran número de escenarios. Es cierto que hay ciertas características de SQL Server que SQL Azure no soporta, pero si soporta todas las más usadas:

* Tablas, tablas temporales, vistas, índices, roles, procedimientos almacenados y funciones.
* Consultas complejas y ‘joins’ entre múltiples tablas.
* Insert, update y delete.
* Restricciones
* Transacciones

Entre las características no soportadas cabe destacar:

* Transacciones distribuidas
* El broker de mensajes de SQL Server
* Consultas a servidores remotos
* Acceso desde tecnologías antiguas, ya obsoletas, en concreto OleDb.

**Figura 3.** Se muestra el panel administrativo de SQL Azure

# 7.5- Bases de conocimiento

Las bases de conocimiento (KB: Knowledge Base) pertenecen a una etapa muy posterior. Surgieron a partir de la investigación en IA como respuesta a las necesidades que las aplicaciones de esta disciplina planteaban. Las bases de conocimiento son la evolución lógica de los sistemas de bases de datos tradicionales, en un intento de plasmar no ya cantidades ingentes de datos, sino elementos de conocimiento (normalmente en forma de hechos y reglas) así como la manera en que estos elementos han de ser utilizados. También se les trata de dotar de conocimiento sobre sí mismas, es decir, una KB ha de "saber lo que sabe". Por ejemplo, ante una consulta determinada, una base de datos accederá a los datos almacenados en ella y dará una respuesta afirmativa o negativa, independientemente de que tenga o no la información relevante; en cambio, una KB respondería "sí", "no" o "no lo sé", en el caso de carecer de todos los datos pertinentes a la consulta. (Pérez, s.f.)

Una base de conocimiento, puede almacenar, además de hechos, un conjunto de reglas que se sirven de esos hechos para obtener información que no se encuentra almacenada de forma explícita. El tipo de base de conocimiento al que se dota de una considerable capacidad de deducción a partir de la información que contiene se denomina sistema experto.

En lo que respecta a las bases de conocimiento, la necesidad de una notación precisa para representar el conocimiento se hizo evidente en el ámbito de la IA casi desde el principio, sin duda debido a la experiencia acumulada en el terreno de las bases de datos. Esta notación recibe el nombre de esquema de representación en el entorno de las bases de conocimiento. En este sentido, resulta práctico considerar una base de conocimiento como un modelo de un mundo/empresa/sección de la realidad (Mylopoulos & Levesque 1984). Hemos de considerar el mundo/universo como una colección de individuos o entidades y una colección de relaciones que existen entre esos individuos. La colección de individuos que conforman el universo a representar y las relaciones que éstos mantienen constituye un estado, y puede haber transformaciones de estado que causan la creación o modificación de individuos o de las relaciones entre ellos. Se puede clasificar los esquemas de representación dependiendo de cuál sea el punto de partida:

* Asignación de valores de certeza sobre estados: esquemas de representación lógicos.
* Individuos/relaciones: esquema de representación de redes semánticas;
* Transformaciones de estados: esquemas de representación procedimentales (sistemas productivos).

La utilidad de un determinado esquema de representación se centra en dos aspectos (Obermeier 1989:23):

* Su capacidad o adecuación expresiva, es decir, lo que el sistema puede "entender" o "decir";
* Su eficacia notacional.

## 7.5.1- Tipos de base de conocimiento

Las bases de conocimiento se han clasificado en dos grandes grupos:

Bases de conocimiento legibles por máquinas, diseñadas para almacenar conocimiento en una forma legible por el computador, usualmente con el fin de obtener razonamiento deductivo automático aplicado a ellas. Contienen una serie de datos, usualmente en la forma de reglas que describen el conocimiento de manera lógicamente consistente. Operadores lógicos como Y (conjunción), O (disyunción), condición lógica y negación son utilizada para aumentarla desde el conocimiento atómico. En consecuencia la deducción clásica puede ser utilizada para razonar sobre el conocimiento en la base de conocimiento. Este tipo de bases de conocimiento son utilizadas por la Web semántica

*Bases de conocimiento legibles por Humanos* están diseñadas para permitir a las personas acceder al conocimiento que ellas contienen, principalmente para propósitos de aprendizaje. Estas son comúnmente usadas para obtener y manejar conocimiento explícito de las organizaciones, incluyen artículos, white papers, manuales de usuario y otros. El principal beneficio que proveen las bases de conocimiento es proporcionar medios de descubrir soluciones a problemas ya resueltos, los cuales podrían ser aplicados como base a otros problemas dentro o fuera de la misma área de conocimiento.

El más importante aspecto de una base de conocimiento es la calidad de la información que esta contiene. Las Mejores Bases de Conocimiento tienen artículos cuidadosamente redactados que se mantiene al día, un excelente sistema de recuperación de información (Motor de Búsqueda), y un delicado formato de contenido y estructura de clasificación. Una Base de Conocimiento puede usar una ontología para especificar su estructura (tipos de entidades y relaciones) y su esquema de clasificación. Una ontología, junto con un grupo de instancias de sus clases constituye una Base de Conocimiento.

Determinando qué tipo de información es capturada, y dónde se encuentra la información en una base de conocimiento es algo que es determinado por los procesos que respaldan al sistema. Una estructura robusta de procesos es la columna vertebral de cualquier Base de Conocimiento. Algunas Bases de Conocimiento tienen un componente de inteligencia artificial. Este tipo de Bases de Conocimiento pueden sugerir soluciones a problemas esporádicos en la retroalimentación por el usuario, y son capaces de aprender de la experiencia (sistemas expertos). Representación de Conocimiento, Razonamiento automatizado y argumentación son las áreas activas de la investigación de la inteligencia artificial.5

## 7.5.2- Bases de conocimiento organizado

**¿Qué son?**

Son instrumentos cognitivos que sirven para organizar, clasificar y representar los conocimientos que se construyen durante el aprendizaje. El secreto de que las bases del conocimiento se utilicen como instrumentos cognitivos está en la organización, ya que la información suelta, dispersa, no es conocimiento, pero la información analizada, organizada y clasificada significativamente se convierte en conocimiento, y el programa de ordenador que lo consigue se convierte en instrumento cognitivo.

Las bases del conocimiento constan de uno o más ficheros, cada uno de los cuales contiene información en forma de cadenas de registros que están relacionados con un dominio de contenido o conjunto de objetos. Cada registro se divide en campos que describen la clase o tipo de información contenida dentro.

Los instrumentos primarios son:

* Buscar la información
* Clasificar la información
* Recuperar la información.

Se puede buscar la información deseada, reorganizarla, ascendiendo o descendiendo a uno o más de los campos existentes, y recuperarla según las necesidades del momento.

Se trata de un ambiente de aprendizaje simple y poderoso en el que los estudiantes examinan la evidencia del mundo real y analizan las controversias del mundo científico, literario, de la historia de la humanidad, etc.

Un aprendizaje basado en una organización del conocimiento conseguirá cara al futuro de nuestros alumnos el hábito de aprender a lo largo de la vida con el objetivo de aprender conocimientos, desarrollar competencias y mejorar sus aptitudes.

Es una metodología activa en la que se aprende, se reaprende y se desaprende a lo largo de la vida de la persona.

La creación del conocimiento es un proceso continuo de interacciones dinámicas entre el conocimiento tácito y el conocimiento explícito. Es decir, un conocimiento abierto que se basa en la intercreatividad. La combinación de interactividad+creatividad, es decir, el proceso de hacer cosas y resolver problemas.

Con esto se desarrolla:

* Pensamiento crítico
* Habilidad para resolución de problemas
* Compartir el pensamiento
* Predisposición hacia el trabajo en equipo
* Tener objetivos claros y creativos

# 7.6- Administrador de proyectos

La administración de proyectos es la disciplina de gestionar proyectos exitosamente, la cual puede y debe aplicarse durante el ciclo de vida de cualquier proyecto (Dixon, 2000).

A lo largo de esta sección se abordaran temas relacionados con la administración de proyectos, sus etapas, y otras consideraciones para comprender a grandes rasgos el objetivo de esta disciplina.

## 7.6.1- Definición de Administración de Proyectos

Existen varias definiciones de la Administración de proyectos, a continuación se muestran algunas:

De acuerdo con una enciclopedia en línea, la administración de proyectos es la disciplina que se encarga de definir y alcanzar objetivos optimizando el uso de recursos: tiempo, dinero, la gente, espacio, etc. (Project management., 2005).

Otra definición nos dice que: la administración de proyectos es la forma de planear, organizar, dirigir y controlar una serie de actividades realizadas por un grupo de personas que tienen un objetivo específico; el cual puede ser (crear, diseñar, elaborar, mejorar, analizar, etc.) un problema o cosa (Rodríguez, 2002).

## 7.6.2- Distinción entre una Metodología de Administración de Proyectos y una Metodología de Desarrollo de Software

Es importante establecer una distinción entre una metodología de Administración de proyectos y una metodología de desarrollo de software. La importancia de distinguir entre ambos conceptos radica en que una organización debe contar con una metodología de administración de proyectos consistente a cualquiera que sea la naturaleza del proyecto que se desarrolla. Las diferencias entre ambos conceptos se enlistan en la Tabla 1 (Neville, 2005).

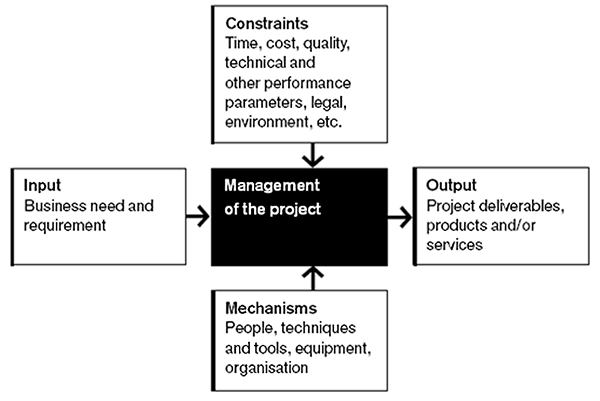
|  |  |
| --- | --- |
| **Metodología de Administración de Proyectos** | **Metodología de Desarrollo de Aplicaciones** |
| Dice que los proyectos deben ser divididos en fases y antes de iniciar con cada una de ellas debe existir un plan | Establece cuáles son las fases y qué actividades involucra |
| Define roles y responsabilidades | Define cuáles son los roles y responsabilidades que corresponden a cada fase |
| Dice que un pre supuesto debe ser definido y administrado. | Define qué medidas deben emplearse para contabilizar el desarrollo en la organización. |

**Tabla 1.** Diferencia entre Metodologías de Administración de Proyectos y de Desarrollo de Aplicaciones. Fuente: (Neville, 2005).

El proceso de administración de proyectos recibe como entradas o es afectado por (Dixon, 2000):

* Necesidades y requerimientos del proyecto a desarrollar (alcance).
* Límites establecidos en tiempo, costo, calidad, desempeño requerido, aspectos legales, etc.
* Mecanismos para lograrlo entre los que están: personas, técnicas, herramientas, equipo y organización.

En base al desempeño de dicha Administración se entregan productos o servicios como salida (Ver Figura 4).



**Figura 4.** Proceso de Administración de Proyectos. Fuente: (Dixon, 2000)

## 7.6.3- Fases de la administración de Proyectos

En la perspectiva tradicional, es posible distinguir 5 componentes de un proyecto (4 etapas más el control) en el desarrollo de un proyecto (Project management. 2005):

* Iniciación de proyecto
* Planificación de proyecto
* Producción de proyecto o ejecución
* Supervisión y control del Proyecto
* Finalización de proyecto o cierre

No todos los proyectos visitarán cada etapa ya que los proyectos pueden ser terminados antes de que alcancen la finalización. Algunos proyectos probablemente no tienen la planificación y/o el control. Y algunos proyectos pasarán por pasos 2, 3 y 4 varias veces (Project management. 2005).

## 7.6.4- Fases del desarrollo de Software

Por otra parte, una de varias metodologías disponibles para el desarrollo de sistemas se compone por las siguientes fases (Rodríguez, 2002):

* Investigación Preliminar
* Diseño del Sistema
* Desarrollo de Sistemas
* Pruebas del Sistema
* Implantación y evaluación

Muchas metodologías varían en las fases que involucran y pueden realizar procesos de retroalimentación o iterativos hasta lograr tener un producto de software terminado. Sea cual sea la metodología, finalmente se busca producir una aplicación lista para su uso.

## 7.6.5- Éxito y fracaso en proyectos de desarrollo de software

Sin considerar el tamaño del proyecto, su alcance o duración existen 5 máximas de satisfacción en su desarrollo (Boyd, 2001):

* Entregar el producto que el cliente desea o necesita
* Entregar la calidad de manera acorde con el precio
* Entregar el producto en el espacio de tiempo que el cliente desea o necesita
* Entregar el nivel de retroalimentación que el cliente desea, y
* Contar con un sistema de resolución de conflictos justo para el cliente y el equipo de desarrollo

Los que se consideran como pasos básicos esenciales para lograr una administración eficiente de proyectos se enumeran a continuación (Toledo, 2002):

Por otro lado los aspectos críticos que contribuyen al fracaso de proyectos de tecnologías de información incluyen (Brock, Hendricks, Linnell, & Smith, 2003) :

* Nunca iniciar sin un objetivo bien definido
* Fragmentar el proyecto
* Invertir tiempo en la planeación
* Involucrar al equipo de trabajo en la planeación y el control
* Fomentar la cohesión del equipo de trabajo
* Prevenir problemas
* Antes de ejecutar, establecer líneas de Base.
* Mantener claro el objetivo principal del proyecto.
* Establecer un proceso para monitorear y controlar
* Atender los puntos críticos primordialmente.
* Tomarse el tiempo necesario para cerrar el proyecto.
* Utilizar una metodología para todos los proyectos.
* Falta de visión clara y establecimiento adecuado de requerimientos
* Expectativas irreales
* Falta de descomposición del proyecto
* Políticas inadecuadas de selección de personal y conflictos en el equipo de desarrollo.
* Falta de involucramiento y enfoque hacia el cliente
* Falta de enfoque estratégico y apoyo administrativo

Tanto los pasos básicos para realizar una administración proyecto como los aspectos críticos que provocan fracasos en este proceso de desarrollo deben de ser contemplados y corregidos, de acuerdo a la forma de trabajar de la empresa. De esta manera se busca mejorar el proceso de desarrollo de los proyectos de software y tomar en cuenta aquellas fallas comunes.(UNAM, s.f.)

# 7.7- SCRUM

Scrum es un proceso en el que se aplican de manera regular [un conjunto de buenas prácticas](http://www.proyectosagiles.org/fundamentos-de-scrum) para ***trabajar colaborativamente, en equipo***, y obtener [el mejor resultado posible](http://www.proyectosagiles.org/beneficios-de-scrum) de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un [estudio de la manera de trabajar de equipos altamente productivos](http://www.proyectosagiles.org/historia-de-scrum).

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en **entornos complejos**, donde se necesita **obtener resultados pronto**, donde los **requisitos son cambiantes o poco definidos**, donde la **innovación**, la **competitividad**, la **flexibilidad** y la **productividad** son fundamentales.

Scrum también se utiliza para resolver situaciones en que **no se está entregando al cliente lo que necesita**, cuando **las entregas se alargan demasiado**, **los costes se disparan** o **la calidad no es aceptable**, cuando se necesita **capacidad de reacción ante la competencia**, cuando **la moral de los equipos es baja y la rotación alta**, cuando es necesario **identificar y solucionar ineficiencias sistemáticamente** o cuando se quiere trabajar utilizando un **proceso especializado en el desarrollo de producto**.

## 7.7.1- Beneficios de SCRUM

Los principales beneficios que proporciona Scrum son:

* Entrega mensual (o quincenal) de resultados (los requisitos más prioritarios en ese momento, ya completados) lo cual proporciona las siguientes ventajas:
  + Gestión regular de las expectativas del cliente y basada en resultados tangibles.
  + Resultados anticipados (time to market).
  + Flexibilidad y adaptación respecto a las necesidades del cliente, cambios en el mercado, etc.
  + Gestión sistemática del Retorno de Inversión (ROI).
  + Mitigación sistemática de los riesgos del proyecto.
  + Productividad y calidad.
  + Lineamiento entre el cliente y el equipo de desarrollo.
  + Equipo motivado.

## 7.7.2- Cómo Scrum proporciona estos beneficios

A continuación se detalla de qué manera Scrum permite conseguir cada uno de los beneficios anteriores:

* **Gestión regular de las expectativas del cliente**

El cliente establece sus expectativas indicando el valor que le aporta cada requisito del proyecto y cuando espera que esté completado. El cliente comprueba de manera regular si se van cumpliendo sus expectativas, da feedback, 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. Se ahorra esfuerzo y tiempo al evitar hipótesis.

* Resultados anticipados (“time to market”)

El cliente puede empezar a utilizar los resultados más importantes del proyecto antes de que esté finalizado por completo.

Siguiendo la ley de Pareto (el 20% del esfuerzo proporciona el 80% del valor), el cliente puede empezar antes a recuperar su inversión (y/o autofinanciarse) comenzando a utilizar un producto al que sólo le faltan características poco relevantes, puede sacar al mercado un producto antes que su competidor, puede hacer frente a urgencias o nuevas peticiones de clientes, etc.

* Flexibilidad y adaptación

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.

Al final de cada iteración el cliente puede aprovechar la parte de producto completada hasta ese momento para hacer pruebas de concepto con usuarios o consumidores y tomar decisiones en función del resultado obtenido.

* Retorno de inversión (ROI)

De manera regular, el cliente maximiza el ROI del proyecto. Cuando el beneficio pendiente de obtener es menor que el coste de desarrollo, el cliente puede finalizar el proyecto.

* Mitigación de riesgos

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. "Si hay que equivocarse o fallar, mejor hacelo lo antes posible". El feedback temprano permite ahorrar esfuerzo y tiempo en errores técnicos.

La cantidad de riesgo a que se enfrenta el equipo está limitada a los requisitos que se puede desarrollar en una iteración. La complejidad y riesgos del proyecto se dividen de manera natural en iteraciones.

* Productividad y calidad

De manera regular el equipo va mejorando y simplificando su forma de trabajar. Los miembros del equipo sincronizan su trabajo diariamente y se ayudan a resolver los problemas que pueden impedir conseguir el objetivo de la iteración. La comunicación y la adaptación a las diferentes necesidades entre los miembros del equipo son máximas (se van ajustando iteración a iteración), de manera que no se realizan tareas innecesarias y se evitan ineficiencias. Las personas trabajan más enfocadas y de manera más eficiente cuando hay una fecha límite a corto plazo para entregar un resultado al que se han comprometido. La consciencia de esta limitación temporal favorece la priorización de las tareas y fuerza la toma de decisiones.

Las iteraciones (Sprints) son regulares y de un mes para facilitar la sincronización sistemática con otros equipos, con el resto de la empresa y con el cliente.

El equipo minimiza su dependencia de personas externas para poder avanzar (depender de la disponibilidad de otros puede parar tareas). La estimación de esfuerzo y la optimización de tareas para completar un requisito es mejor si la realizan las personas que van a desarrollar el requisito, dadas sus diferentes especializaciones, experiencias y puntos de vista. Asimismo, con iteraciones cortas la precisión de las estimaciones aumenta. Las personas trabajan de manera más eficiente y con más calidad cuando ellas mismas se han comprometido a entregar un resultado en un momento determinado y deciden cómo hacerlo, no cuando se les ha asignado una tarea e indicado el tiempo necesario para realizarla.

* Alineamiento entre cliente y equipo

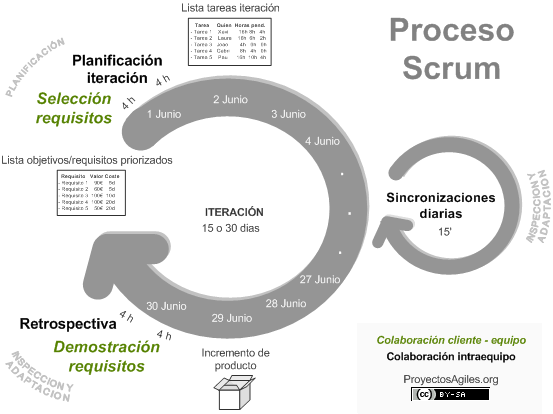
Los resultados y esfuerzos del proyecto se miden en forma de objetivos y requisitos entregados al negocio. Todos los participantes en el proyecto conocen cuál es el objetivo a conseguir. El producto se enriquece con las aportaciones de todos.

* Equipo motivado

Las personas están más motivadas cuando pueden usar su creatividad para resolver problemas y cuando pueden decidir organizar su trabajo. Las personas se sienten más satisfechas cuando pueden  mostrar los logros que consiguen.

## 7.7.3- El proceso SCRUM

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes natural y hasta de dos semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. (Ver Figura 5)



**Figura 5.** Proceso SCRUM

El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente prioriza los objetivos balanceando el valor que le aportan respecto a su coste y quedan repartidos en iteraciones y entregas. De manera regular el cliente puede maximizar la utilidad de lo que se desarrolla y el retorno de inversión mediante la re planificación de objetivos del producto, que realiza durante la iteración con vista a las siguientes iteraciones.

Las actividades que se llevan a cabo en Scrum son las siguientes:

**Planificación de la iteración**

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

1. *Selección de requisitos (4 horas máximo).* El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que se compromete a completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.
2. *Planificación de la iteración (4 horas máximo).* El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se auto asignan las tareas.

**Ejecución de la iteración**

Cada día el equipo realiza una reunión de sincronización (15 minutos máximo). Cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con el compromiso adquirido. En la reunión cada miembro del equipo responde a tres preguntas:

1. ¿Qué he hecho desde la última reunión de sincronización?
2. ¿Qué voy a hacer a partir de este momento?
3. ¿Qué impedimentos tengo o voy a tener?

Durante la iteración el Facilitador (Scrum Master) se encarga de que el equipo pueda cumplir con su compromiso y de que no se merme su productividad.

Elimina los obstáculos que el equipo no puede resolver por sí mismo.

Protege al equipo de interrupciones externas que puedan afectar su compromiso o su productividad.

**Inspección y adaptación**

El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

1. *Demostración (4 horas máximo).* El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, re planificando el proyecto.
2. *Retrospectiva (4 horas máximo).* El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

# 8.- Procedimiento y descripción de las actividades

**8.1- Análisis**

La identificación del problema se hizo mediante la solicitud del director del departamento de software de la empresa Massoluciones y de acuerdo a las necesidades de cambio.

Durante la estadía en las residencias se desempeñan actividades ajenas al proyecto lo cual es un beneficio para poder socializar y conocer más sobre la empresa.

Se sabe que se requiere un software el cual funja el papel de base de conocimiento y además sea un administrador de proyectos para la empresa. Lo que se tiene muy en claro es que dicho software deberá cumplir estos requisitos por igual, siendo esto primordial, además de que toda la información valiosa generada en dicho software deberá pertenecer únicamente a la empresa.

**8.2- Personas interesadas**

* Director del departamento de Software

**8.3- Requerimientos funcionales**

* El software deberá contar con autenticación de usuarios.
* El software deberá fungir como base de conocimiento y almacenar toda la información valiosa de la empresa.
* El software deberá fungir como administrador de proyectos para la empresa.
* El software podrá asignar diferentes permisos a los usuarios registrados mediante un usuario administrador.
* Los usuarios registrados podrán generar la información valiosa de la empresa así como su consulta.
* El software permitirá agregar y eliminar usuarios.

**8.4- Requerimientos no funcionales**

* El software deberá siempre estar disponible para las consultas de información.
* El software deberá ejecutarse en cualquier computadora.

**8.5- Planeación e investigación**

Una vez que se tenían los requerimientos claros se realizaron una serie de juntas con el equipo de desarrollo y en equipos de pares se dieron tareas diversas de investigación para determinar el mejor software que se adaptara a los requerimientos del director del departamento.

La serie de juntas e investigaciones arrojo como resultado un Software llamado *Confluence* (Ver Figura 6)*,* el cual es un software de colaboración en equipo y fue desarrollado y distribuido como un software como servicio, bajo una renta mensual por la empresa Atlassian.

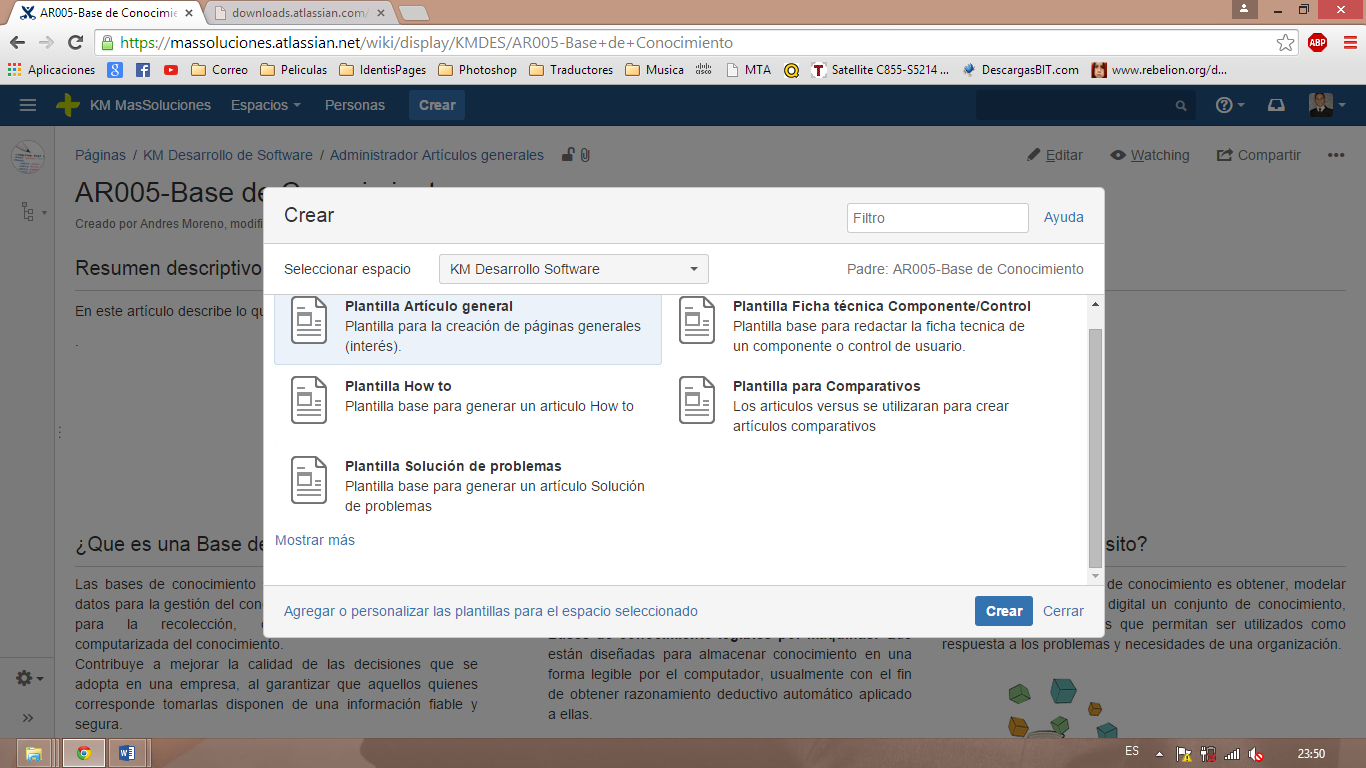
**Figura 6.** Logotipo de Confluence y Atlassian

A partir de este momento y una vez definido el software se dio comienzo a generar la información valiosa de la empresa, hasta este punto se le asignó esta tarea a un servidor. De esta forma el primer paso que se dio después de la elección del software fue investigar más acerca del mismo, entender su estructura, su funcionamiento.

Para comenzar Confluence maneja diferentes tipos de artículos para la base de conocimiento los cuales son (Ver Figura 7):

* Artículos *How to*: Son artículos que guían al usuario mediante una serie de pasos a lograr un objetivo o tarea *Ejemplo: Como replicar una base de datos*.
* Artículos *Troubleshooting*:Este tipo de artículos ofrecen una solución a un problema específico del usuario *Ejemplo: Solución al error de idioma en la instalación de SQL Server 2008*.
* Artículos *Versus*: Estos artículos versus se utilizan para crear artículos comparativos.

Además de estos artículos base Confluence ofrece la posibilidad de crear plantillas de artículos propias, a medida de que la empresa las requiera.

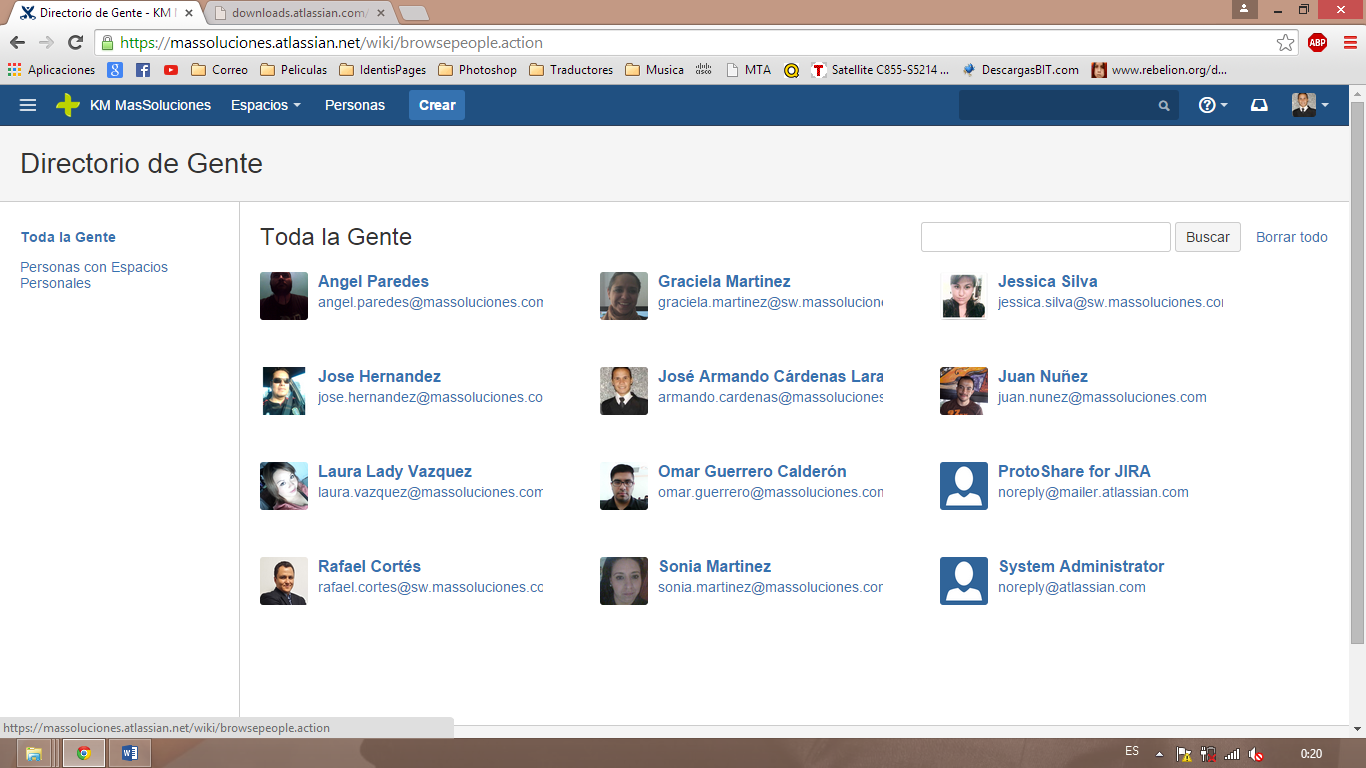


**Figura 7.** Se muestran los diferentes tipos de plantillas para artículos.

Además de estas plantillas para generar los artículos y la información valiosa, Confluence se basa en la creación y uso de espacios (Ver Figura 7).

Estos espacios son contenedores de páginas y publicaciones en blogs con contenido relacionado, cada usuario tiene un espacio personal en el que puede crear entradas de blog (personal) y agregar artículos a la base de conocimiento (páginas de información valiosa). Dichos espacios vienen en dos variantes como anteriormente se describió:

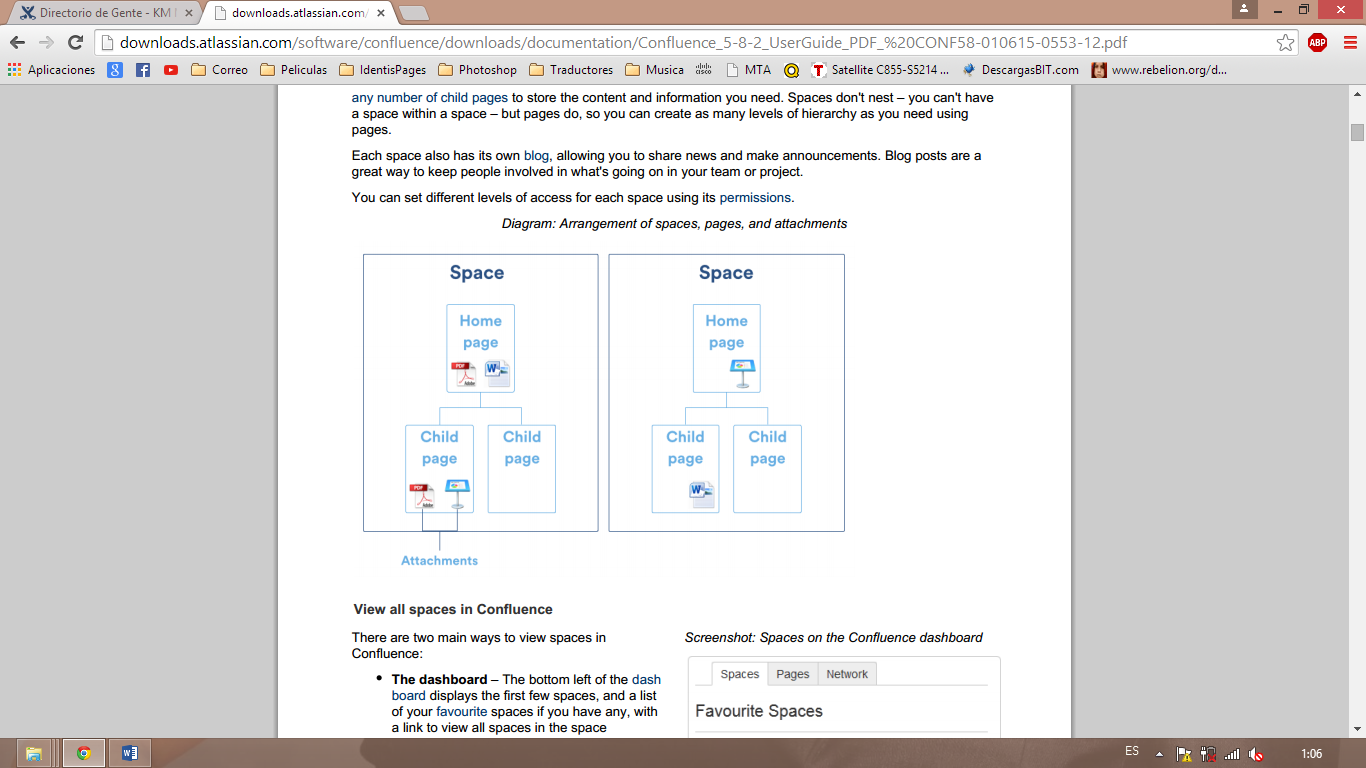
* **Espacios de sitio:** Algunas veces llamados sitios “Globales” en donde el usuario puede generar los artículos de información valiosa para la empresa y contribuir con sus compañeros de trabajo.
* **Espacios personales:** Donde cualquier usuario registrado puede configurar el perfil personal, con opción a permanecer privado para otros usuarios del software. Estos espacios personales están ubicados en el directorio de personas.(Ver Figura 8)



**Figura 8.** Se muestra el directorio de usuarios en el software Confluence.

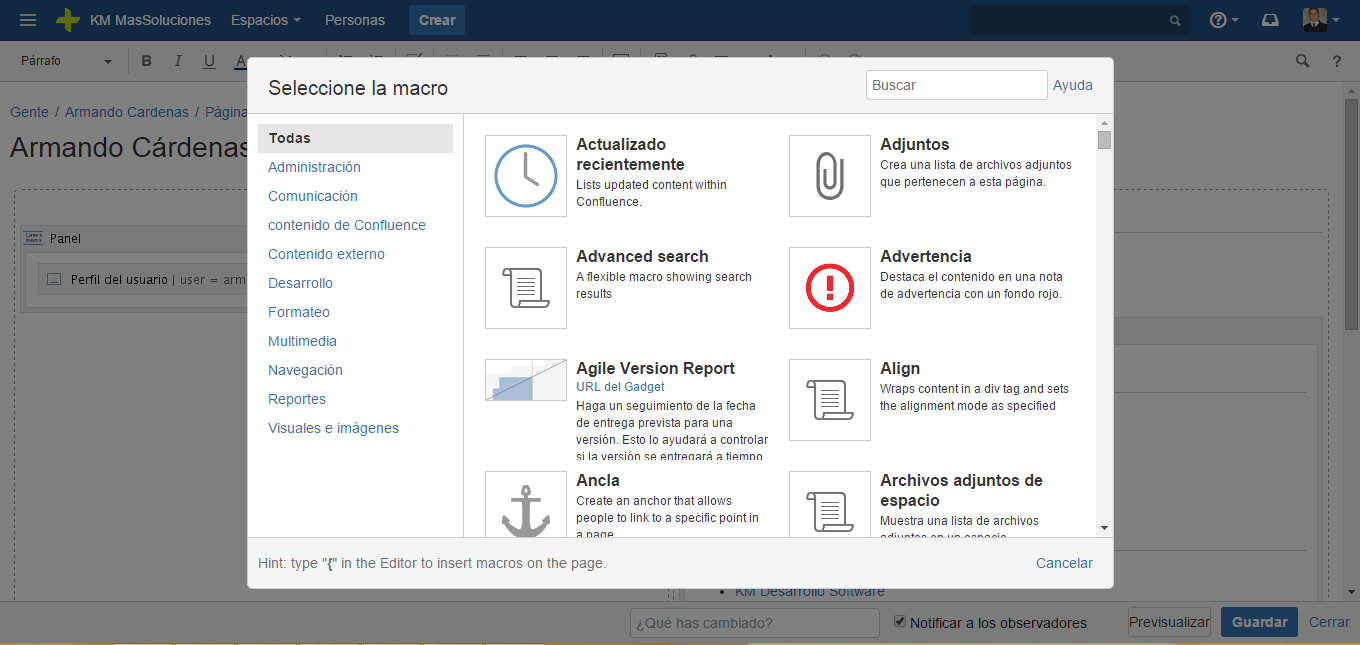
Todos los espacios tienes al menos un administrador, usualmente es el mismo usuario que creo el espacio en cuestión, y es el mismo usuario que puede otorgar los permisos a ese espacio (como crear nuevos artículos páginas), inclusive otorgar permisos administrativos a otros usuarios.

Cada espacio creado, incluyendo el espacio personal crea automáticamente una página de inicio, la cual es la primera página que observa el usuario y que puede editar a su gusto, de esta forma el usuario puede crear cualquier número de páginas hijas para almacenar información. (Ver Figura 9)



**Figura 9.** Ordenamiento de espacios, páginas y adjuntos

Entre las actividades que se encomendaron la primordial y más importante fue la de alimentar la base de conocimiento, ya que este software había sido adquirido bajo el pago de una licencia mensual lo que significaba que se tenía que dar comienzo lo antes posible. Durante las primeras semanas la investigación acerca de este software/plataforma fue ardua y con muy buenos resultados, ya que la edición y creación de páginas y artículos nuevos requerían de un editor (ya integrado en el software) y que contaba con ciertos *Macros* (herramientas de estilo) que daban el formato y estilo a la página. (Ver Figura 10).



**Figura 10.** Se muestran las diferentes macros en la edición de un artículo.

Una vez dominado en gran parte el manejo del software, se planearon juntas informativas para el equipo de desarrollo, explicando y aclarando dudas acerca del mismo, puesto que todo el equipo daría comienzo a alimentar la base de conocimiento también.

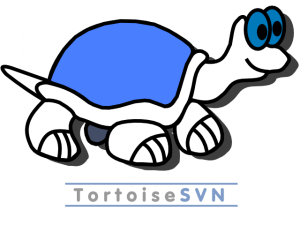
Así fue como la nueva base de conocimiento se fue alimentando de toda la información valiosa que la empresa generaba día con día, el inicio fue complicado ya que ningún integrante del departamento tenía el habito de documentar las experiencias propias y solución a problemas que diariamente se vivían.

A partir de este punto se dio comienzo a una nueva etapa en las residencias profesionales, las auditorias.

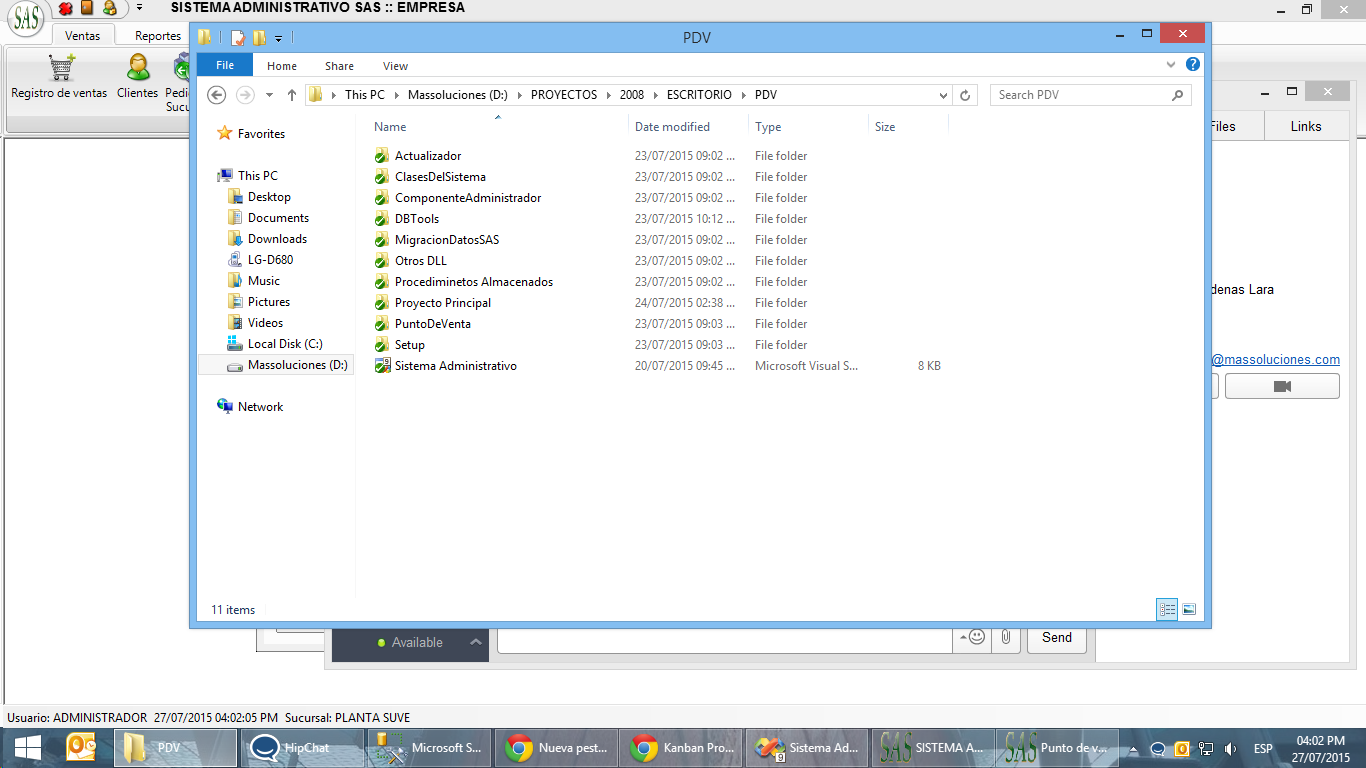
Massoluciones maneja un versionamiento de proyectos en su servidor principal, mediante el software *TortoiseSVN* (Ver Figura 11), en donde cada integrante que participara en el desarrollo de un proyecto debía mantener siempre actualizada la carpeta del proyecto en el servidor, de esta forma los demás integrantes del equipo de desarrollo trabajarían sobre una copia del mismo proyecto.

El problema radicaba en que algunos integrantes del equipo no acostumbraban a versionar sus cambios más recientes en el proyecto (versionar) y esto era causa de problemas más graves, como la unión no intencional de diferentes versiones del proyecto desactualizadas, por esta causa surge la idea de auditar cada equipo del departamento dos días de la semana (lunes y viernes) en donde se debía verificar que cada integrante mantuviera su copia de proyecto siempre versionada.

(Ver Figura 12)



**Figura 11.** Logotipo del software de control de versiones TortoiseSVN

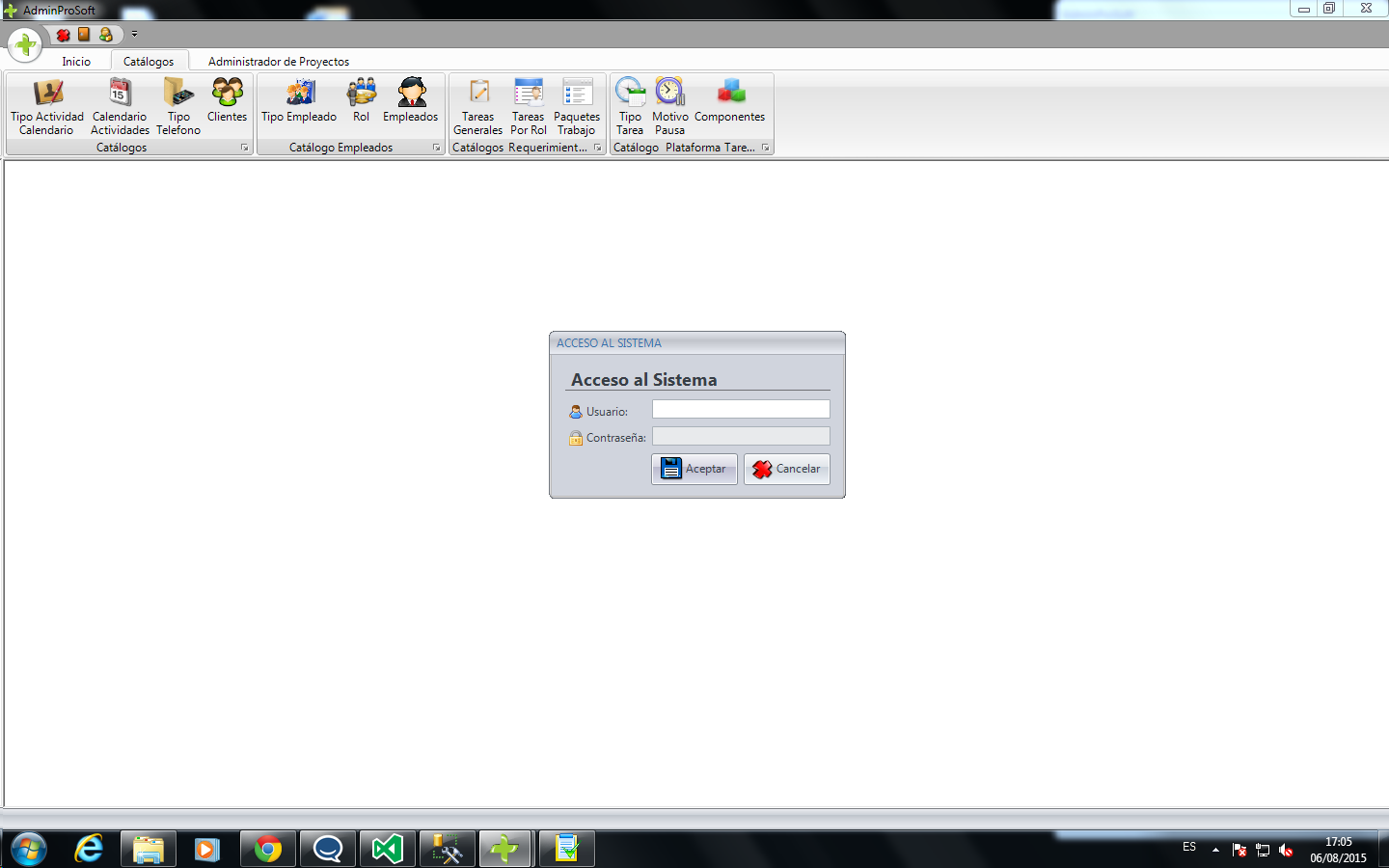


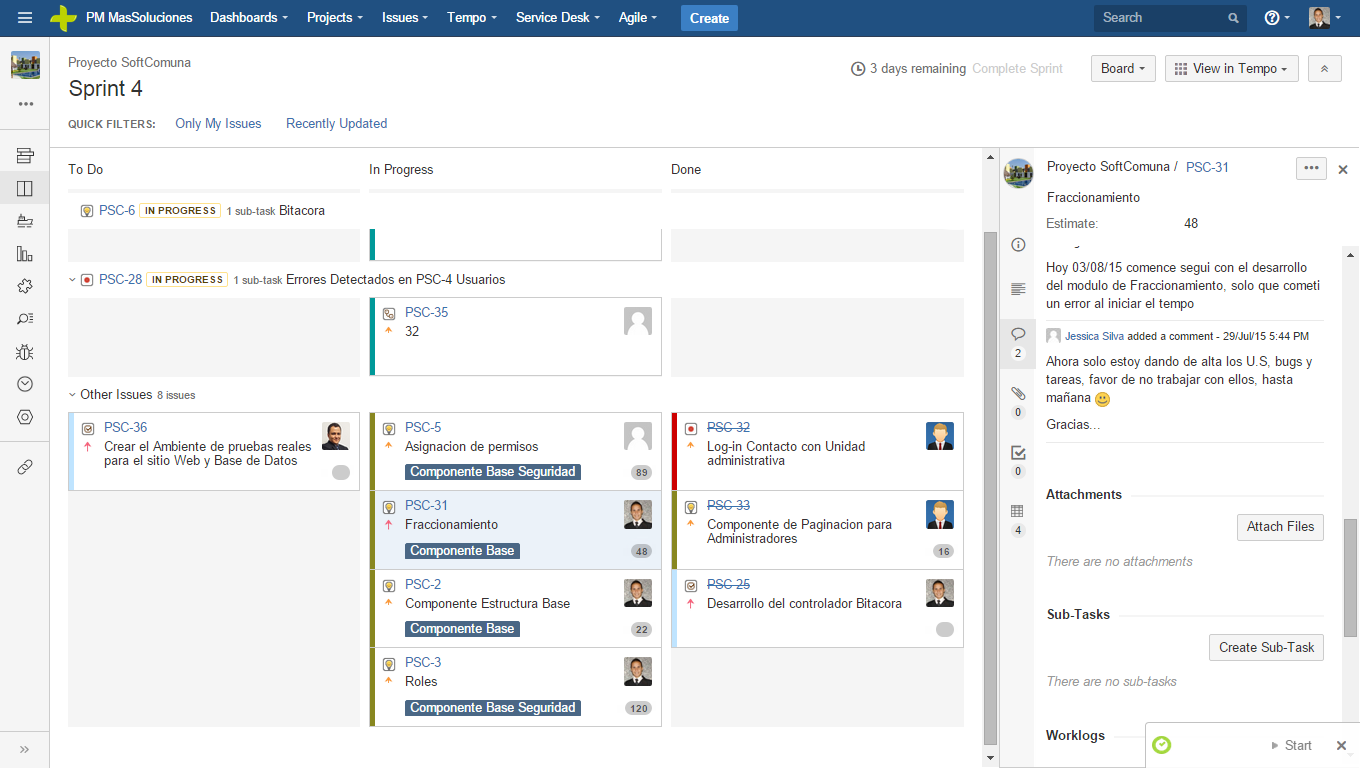
**Figura 12,** Proyecto correctamente versionado en TortoiseSVN

Los resultados de estas auditorías no se vieron reflejados sino hasta alrededor de un mes después, y con muy buenos resultados ya que los versionamiento y habían incrementado antes el servidor principal y los errores decrementado.

A mediados del mes de Mayo se toma la decisión de contratar el servicio Jira para dejar atrás la administración de proyectos mediante el software *Adminprosoft* (Ver Figura 13) ya que este software que originalmente fue desarrollado por la misma empresa no manejaba una metodología de mejora continua, y lo que se buscaba era implementar la administración de los futuros proyectos bajo este esquema.

Por otro lado *Jira* era todo lo contrario ofrecía una planificación en base a prioridades, ofrecía un seguimiento por requisitos y daba la oportunidad de ligar artículos creados en Confluence. Además de todo esto Jira era capaza de asignar tareas específicas a los integrantes del equipo de trabajo y establecer tiempos, los informes que se podían obtener de Jira eran en base a los tiempos, y el rendimiento del equipo. (Ver Figura 14)

**Figura 13.** Software Adminprosoft



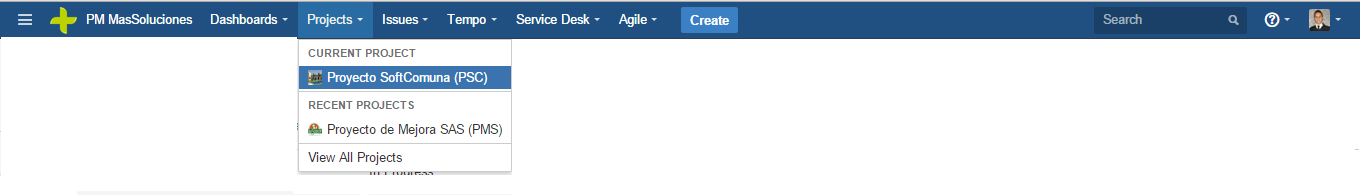
**Figura 14.** Dashboard planificación y desarrollo de actividades y tareas en un proyecto

Jira es manejado en base a proyectos, al igual que Confluence con los artículos, espacios y páginas, de esta forma en Jira pueden ser creados con facilidad nuevos espacios para proyectos (Ver Figura 15).

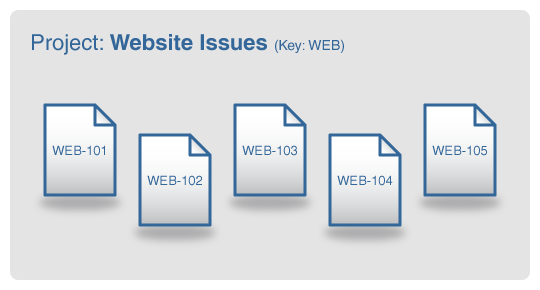
Un proyecto administrado por Jira consta de una colección de *Issues* y es definido de acuerdo a los requerimientos del proyecto

Ejemplos de proyectos en Jira:

* Proyecto de desarrollo de software
* Campaña de marketing
* Sistema de ayuda

Cualquier issue debe pertenecer a un proyecto en específico, y cada proyecto tiene un nombre (Ejemplo: Issues WEB) y su llave seria WEB, de esta forma cada issue que pertenezca a este proyecto tendría la siguiente nomenclatura en su nombre (Ver Figura 16).

**Figura 15.** Ejemplo de los espacios para proyectos en JIRA



**Figura 16.** Nomenclatura de nombre de los issue de un proyecto en JIRA.

Jira administra los proyectos bajo una metodología de mejora continua, y de desarrollo a medida del cliente, el primer proyecto que se comenzó bajo la administración de Jira se llama Softcomuna el cual será un software como servicio enfocado a la administración de fraccionamientos residenciales, bajo una licencia de pago mensual y con soporte técnico especializado. Los resultados han sido bastante buenos comparados con otros desarrollos pasados, gracias a los informes generados por Jira el aprovechamiento del tiempo ha incrementado, hoy en día el desarrollo de este software avanza cada día más, esperando su terminación dentro de los próximos ocho meses. Jira también ofrece distintas herramientas para llevar a cabo la documentación de los proyectos, crear diagramas de clases, de flujo, prototipos, y diseños es muy sencillo además de una gran ayuda, Jira cuenta con un alojamiento de archivos en nube por lo que cualquier documento adjunto a algún issue siempre estará a salvo y siempre disponible.

Softcomuna almacenará su base de datos principal en SQL Azure, mediante un manejo de servidores en nube que facilitan el acceso a la información en cualquier momento del día, lo que hace que el software sea mucho más eficaz ya que no se hará uso de servidores locales en los equipo de cómputo del fraccionamiento cliente.

Softcomuna pretende el primer software ofrecido al mercado como servicio por Massoluciones, de tal forma que todo el desarrollo ha sido bajo un patrón llamado MVC (Model, View, Controller) se tuvo que adaptar ya que los proyectos anteriores eran desarrollados bajo modelo de 3 capas, así que el aprendizaje ha sido en todos los aspectos nuevo, tanto de desarrollo personal, laboral y de conocimiento.

# 9.- Conclusiones

El proyecto Softcomuna está diseñado para cambiar el paradigma de la administración de los fraccionamientos en la ciudad de Celaya, Softcomuna ofrecerá una serie de servicios para los dueños de casas ubicadas en los fraccionamientos, mediante una renta mensual podrá acceder a una red social propia y única para cada fraccionamiento que adquiera la membresía, pagar sus cuotas de mantenimiento median PayPal o tarjetas de crédito, configurar su perfil de usuario y subir fotografías personalizadas.

Este proyecto ha sido planeado con una duración de ocho meses de los cuales han transcurrido dos, cada dos semanas se realiza una junta con el cliente para mostrar los avances logrados y las incidencias presentadas, a cada semana de desarrollo planeada se le da el nombre de Sprint, y la reunión con el cliente es nombrada Sprint review.

En general esta primera versión de Softcomuna no será la definitiva, se tienen planes a futuro para expandir el software y ofrecer más servicios a los administradores de los fraccionamientos. Por lo que las actualizaciones serán constantes

En lo personal mi estancia en la empresa Massoluciones ha sido de las mejores experiencias que he tenido, las residencias profesionales para un estudiante significan el primer paso a la vida laboral, ya no hay presión de tareas, de maestros, de exámenes; pero se da comienzo a otro tipo de presiones puesto que en este caso ya no hay segundas oportunidades, debes dar el cien por ciento de ti, porque el equipo espera que lo hagas.

Recuerdo que en la entrevista que me hicieron en Massoluciones el CEO de la empresa me pregunto que esperaba de las residencias ahí, a lo que yo respondí: “Quisiera aprender, y dar lo mejor de mí para aprender todos al mismo tiempo”, hoy puedo decir que si así fue aprendí demasiado, di y estoy dando lo mejor de mí para cada día superarme.

Hoy en día formalmente estoy integrado al equipo del departamento de software y debo decir que el trabajo en equipo ha sido realmente primordial, tantos profesores que siempre lo comentaron en clases, pero uno como estudiante le cuesta trabajo entenderlo de esa forma ya se necesita vivir la experiencia para decir: “cuanta razón tenían todos lo que me dijeron lo mismo”.

La documentación fue uno de los temas que más me costó trabajo implementar y que a la fecha aún estoy en proceso de dominar, pero que gracias a mis compañeros no me ha sido tan difícil como yo esperaba.

En la escuela te enseñan las técnicas, las maneras de realizar alguna tarea, pero la realidad es que la experiencia la obtienes una vez que estas fuera de ese ambiente estudiantil, y es un gran paso de más de treinta de mis compañeros de grupo que iniciamos en el primer semestre no pudieron dar, me siento orgulloso de estar donde ahora estoy, de saber que estoy a nada de ser completamente un egresado, me siento orgulloso de estar a un paso de tocar la mítica campana aquella campana que tantas veces escuche sonar, me siento orgulloso de ser…¡ORGULLOSAMENTE LINCE!

# Referencias bibliográficas

Castro, L. (s.f.). *About*. Obtenido de http://aprenderinternet.about.com/od/Glosario/g/Software-como-servicio.htm

Culebro Juárez , M., Gómez Herrera, W., & Torres Sáchez, S. (2006). *Software libre vs Software propietario Ventajas y Desventajas.* Mexico.

EMC2. (s.f.). *http://mexico.emc.com/*. Obtenido de http://mexico.emc.com/corporate/glossary/software-as-a-service.htm

MasterMagazine. (s.f.). *MasterMagazine*. Obtenido de http://www.mastermagazine.info/termino/6751.php

Nube, E. e. (s.f.). *Estoy en la Nube*. Obtenido de http://www.estoyenlanube.com/recursos/sql-azure/que-es-sql-azure/

Pérez, C. (s.f.). *http://elies.rediris.es/*. Obtenido de http://elies.rediris.es/elies18/522.html

Pressman, R. S. (2010). *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO.* Mexico: McGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V.

Ramírez, G. K. (2006). *Software libre vs software propietario.* Mexico.

Stallman, R. M. (2004). *Software libre para una sociedad libre.* Madrid: Traficantes de Sueños.

UNAM, R. (s.f.). *http://www.revista.unam.mx/*. Obtenido de http://www.revista.unam.mx/vol.7/num6/art47/art47.htm