CONSTRUIR UN SISTEMA DE INFORMACIÓN PARA LA TIENDA LA LOMITA

Oscar Albeiro Castillo Alarcón

Johan Fernando Chitiva Garzón

Daniel Alejandro Macías Silva

Leidy Tatiana Roa Miguez

Laura Alejandra Villamil Ruiz

PROPUESTA DE TRABAJO DE GRADO PARA OPTAR AL TÍTULO DE TECNÓLOGO EN ADSI

ING. Frank Castillo

SERVICIO NACIONAL DE APRENDIZAJE (SENA)

Centro de Diseño y Metrología ADSI

Bogotá

06 de Agosto de 2015

**CONTENIDO**

**TEMA PAG**

INTRODUCCIÓN………………………………………………………………… 1

DESCRIPCIÓN DEL PROBLEMA……………………………………………… 2

JUSTIFICACIÓN INICIAL O PRELIMINAR……………………………………. 3

OBJETIVOS……………………………………………………………………….. 4

METODOLOGICA KENDALL Y KENDALL……………………………………. 5

RECOMENDACIONES………………………………………………………….. 7

ALCANCE DEL PROYECTO……………………………………………………. 8

RECURSOS………………………………………………………………………. 9

PSP………………………………………………………………………………… 10

TSP…………………………………………………………………………………. 14

METODOLOGIAS…………………………………………………………………..17

DIAGRAMA DE GANTT…………………………………………………………….21

DIAGRAMA DE PERT.…………….…………………………….…………………23

BIBLIOGRAFÍA…………………………………………………………………….. 24

**INTRODUCCIÓN**

En el siguiente trabajo vamos a observar la propuesta de trabajo del proyecto de la tienda la lomita, la cual busca crear un sistema de información para que esta tenga un mejor orden en su inventario cartera y flujo de dinero, además de conocer los conceptos de que es tsp y psp y para qué sirven cada uno y las diferentes metodologías que hay para desarrollar proyectos.

1

**1. DESCRIPCIÓN DEL PROBLEMA (Tema de investigación).**

El problema de nuestro proyecto es que la tienda la lomita no lleva un control en el inventario de sus productos ni tampoco en el flujo de dinero de la misma. Debido a que La dueña no lleva un control y lo que hace es anotar o apuntar en un cuaderno lo que vende; pero no sabe si está ganando o perdiendo más dinero en la tienda y para solucionar esto vamos a usar los siguientes requerimientos funcionales y los requerimientos no funcionales.

Requerimientos funcionales

* Hacer que se lleve un control del dinero que ingresa a diario a la tienda
* Utilizar un método de control para saber cuánto se gasta en la compra de los productos
* Hacer un inventario de los productos de la tienda
* Llevar un orden de los proveedores tiene la tienda
* Saber cuántos productos sobraron en el mes
* Llevar un orden y control de que producto se fía y a quien se le fía
* Tener un control de la parte monetaria que ingresa y sale de la tienda.
* Usar el método de cartera.

Requerimientos no funcionales

* Rendimiento
* Disponibilidad
* Seguridad
* Accesibilidad
* Usabilidad
* Estabilidad
* Operatividad
* Estabilidad
* Concurrencia

2

**2. JUSTIFICACIÓN INICIAL O PRELIMINAR.**

Este proyecto lo realizamos con el propósito de dar una solución al problema que tiene la tienda la lomita. El cuál es la falta de control en el inventario de sus productos y el flujo de dinero, para ello le sugerimos al cliente una arquitectura de hardware de 32 bits y una arquitectura de software para PC porque al ser una sola tienda no hay necesidad de tener más computadores para llevar el registro y no necesitamos trasportar o pasar la información a otros computadores y que tenga un sistema operativo Windows 7 ultímate.

Le ofrecemos estas arquitecturas debido a que:

* Tiene un computador el cual ya tiene instalado Windows 7 ultímate de 32 bits por lo tanto podemos usar ese, además de que se busca la economía del cliente y lo que le brindamos le sea útil y fácil de manejar.
* Además si le sugerimos como sistema operativo un Linux o un Mac os le saldría más caro y la solución debe ser eficaz y económica además de que debemos aprovechar lo que ya se tiene.

3

**3. OBJETIVOS**

**OBJETIVO GENERAL.**

Controlar y organizar el inventario de los productos, la cartera y el flujo de dinero de la tienda por medio de un sistema de información para escritorio

**OBJETIVOS ESPECÍFICOS.**

* Controlar el inventario, la cartera y el flujo de dinero.
* Realizar una investigación de mercado con una duración de 3 semanas en la cual se les aplique una encuesta al cliente.
* Usar los requerimientos que necesita el cliente con el fin de satisfacer sus necesidades y poder dar a conocer el resultado.
* Usar la recolección de datos como un medio para obtener más información acerca de la tienda.
* Aplicar los conocimientos adquiridos en el centro de diseño y metrología en el programa de formación (ADSI)

**4**

4. **METODOLOGICA KENDALL Y KENDALL.**

1. **Identificación de problemas, oportunidades y objetivos:**Esta fase es crucial para el éxito del resto del proyecto requiere que se observe de forma objetiva lo que ocurre en una organización, luego en conjunto con otros miembros de la organización hacer notar los problemas. Las oportunidades son aquellas situaciones que se considera que pueden mejorarse, perfeccionarse mediante el uso de los sistemas de información. También es un componente importante de la primera fase, en esta etapa se deberá descubrir lo que la organización intenta realizar, luego determinar si el uso de los sistemas de información apoyaría a la organización para alcanzar sus metas.
2. **Determinación de los requerimientos de información:** Esto se hace a partir de los usuarios particularmente involucrados, para determinar los requerimientos de información dentro de una organización pueden utilizarse diversos instrumentos, los cuales incluyen: muestreo, el estudio de los datos y formas usadas para la organización, la entrevista, los cuestionarios; la observación de la conducta de quien tomo la decisiones, así como de su ambiente. Se hace todo lo posible por identificar qué información requiere el usuario para desempeñar sus tareas.
3. **Análisis de las necesidades del sistema:** Se analizan las necesidades propias del sistema, para ello existen herramientas y técnicas diseñadas para tal fin, estas incluyen entre otras el uso de los diagramas de flujo de datos que cuentan con una técnica estructurada para representar en forma gráfica la entrada de datos a la organización, los procesos la salida de información. También se analizan las decisiones estructuradas por realizar, que son decisiones donde las condiciones, condiciones alternativas, acciones y reglas de acción podrán determinarse.
4. **Diseño del sistema recomendado**: Se usa la información recolectada con anterioridad y se elabora el diseño lógico de sistemas de información, se diseña también procedimiento es precisos de captura de datos, con la finalidad de que los datos que se introducen en el sistema de información, sean los correctos. Esta etapa también incluye el diseño de los archivos o la base de datos que almacenará aquellos datos requeridos por quien toma las decisiones en la organización.

**5**

1. **Desarrollo y documentación del software**: Dentro de las técnicas estructuradas para el diseño y documentación del software se tienen: el método HIPO, los diagramas de flujo, los diagramas Nassi.Schneiderman, los diagramas Warnier-Orr y el pseudocódigo es aquí donde se transmite al programador los requerimientos de programación.
2. **Pruebas y mantenimiento del sistema:** Todo sistema de información debe probarse antes de ser utilizado, ya que el costo es menor si se detectan los problemas antes de que entre en funcionamiento. En un principio, se hace una serie de pruebas, con datos tipo, para identificar las posibles fallas del sistema, más adelante, se utilizarán los datos del sistema real.
3. **Implantación y evaluación del sistema:**Esta es la última etapa del desarrollo del sistema, esto incluye el adiestramiento que el usuario requerirá. Aunque la evaluación del sistema se plantea como parte integrante de la última etapa del ciclo de desarrollo de los sistemas; realmente la evaluación toma parte de cada una de las etapas. Uno de los criterios fundamentales que debe satisfacerse, es que el futuro usuario utilice el sistema desarrollado.

**6**

**RECOMENDACIONES**

1. Planificar cualquier actividad preventiva a partir de un análisis de necesidades elaborado con participación de todos los implicados.
2. Promover programas de gestión por objetivos como una de las formas más efectivas.
3. Llevar a cabo sistemáticamente actividades de evaluación del esfuerzo preventivo y no sólo de los resultados.

**7**

**ALCANCE DEL PROYECTO**

El alcance de nuestro proyecto es implementar un sistema de información, el cual ayude a la tienda la lomita teniendo como excepción que el sistema de información que vamos a implementar, sería solo para la tienda la lomita y sus clientes, ubicada en el barrio Alfonso López en la localidad quinta de Usme, lo que buscamos con este sistema de información es mejorar el control y la organización del inventario, el flujo de dinero y la cartera teniendo siempre como objetivo principal dar solución al problema que tiene la tienda la lomita .

**8**

**RECURSOS**

|  |  |
| --- | --- |
| **RECURSOS HUMANOS** | |
| **Instructores:** | **Aprendices:** |
| Frank castillo | Oscar castillo |
| Oscar Benavides | Johan Chitiva |
| Zulma espinel | Daniel Macías |
| Jorge prieto | Leidy roa |
| Enrique prieto | Laura Villamil |

|  |  |
| --- | --- |
| **RECURSOS MATERIALES E INFRAESTRUCTURA.** | |
| **Materiales**: | **Infraestructura:** |
| Computador | Servicio Nacional de Aprendizaje ( Sena) |
| Internet | Tienda la lomita |
| Libros |
| Cuadernos |

**9**

**PSP**

**Que es psp:**

El Personal Software Process, conocido por sus siglas como PSP, es una metodología de reciente creación, proveniente del Instituto de Ingeniería del Software (SEI). PSP es una alternativa dirigida a los ingenieros de sistemas, que les permite mejorar la forma en la que construyen software. Considerando aspectos como la planeación, calidad, estimación de costos y productividad, PSP es una metodología que vale la pena revisar cuando el ingeniero de software está interesado en aumentar la calidad de los productos de software que desarrolla dentro de un contexto de trabajo individual.

**CARACTERISTICAS**

En PSP todas las tareas y actividades que el ingeniero de software debe realizar durante el proceso de desarrollo de un producto de software, están puntualmente definidas en un conjunto de documentos conocidos como scripts. Los scripts son el punto medular de PSP, por lo que se hace mucho énfasis en que deben ser seguidos en forma disciplinada, ya que de ello dependerá el éxito de la mejora que se busca. Gran parte de las tareas y actividades definidas en los scripts generará en su realización un conjunto de datos, fundamentalmente de carácter estadístico. La aplicación de PSP en varios procesos de desarrollo, y el análisis de la información estadística generada en cada uno de éstos, permitirán al ingeniero de software identificar, tanto sus fortalezas como sus debilidades, y crecer a través de un proceso de auto aprendizaje y auto mejora.

La calidad en PSP, es un aspecto fuertemente relacionado con la cantidad de defectos que el producto de software contiene. En este nivel se introducen algunos métodos aplicables al proceso de desarrollo de software, dentro de un enfoque de proyectos a gran escala, pero sin lidiar con problemas de comunicación y coordinación de los equipos de trabajo

**10**

**VENTAJAS Y DESVENTAJAS PARA UTILIZAR PSP**

PSP es una alternativa, de las muchas que han surgido recientemente, para mejorar el proceso de desarrollo de software. Más que clasificar un conjunto de sentencias como ventajas o desventajas:

* PSP es una metodología basada en estimación. La estimación permite saber cuándo y cómo se desarrollan las tareas de un proceso, por lo que podría citarse como un aspecto importante de esta metodología el estar basada en métricas y estimaciones.
* La información de las métricas y estimaciones se utiliza para evaluar y mejorar procesos futuros. PSP parte de la premisa que, si el ingeniero de software conoce sus fortalezas y debilidades, puede establecer las acciones necesarias para erradicar o explotar los aspectos identificados en la forma en que desarrolla software.
* Aunque lo mencionado en el punto anterior podría sonar bastante atractivo, la forma de llegar a ese auto conocimiento puede resultar tediosa y, en el peor de los casos, una pesadilla para el desarrollador. Salvo muy pocas excepciones, los ingenieros de software nunca realizan procedimientos formales para conocer la forma en que trabajan, no saben con exactitud cuántas líneas de código generan por hora, cuánto tiempo invierten al corregir un error, cuánto tiempo invierten en pruebas, etcétera.
* Los pasos de registro de información a detalle en el nivel de medición pueden resultar frustrantes cuando se tiene presión de tiempo.
* En los scripts de PSP no se incluyen tareas y actividades para la etapa de análisis de requerimientos. Siempre se parte de una definición de requerimientos que no va a cambiar.

**11**

**PASOS PARA IMPLANTACION PSP**

* Los ingenieros deben ser entrenados por un instructor calificado de PSP.
* La Capacitación es sobre grupos o equipos, y serán grupos que así lo han sido y seguirán siendo.
* Requiere un fuerte soporte de administración, en este sentido es necesario que los administradores entiendan el PSP, saber cómo apoyarlos y como monitorear sus avances, sin un adecuado monitoreo los ingenieros caerán otra vez en los malos hábitos.
* Después de ser bien entrenados y bien administrados lo que sigue es optimizar la interacción entre equipos y aquí entraría Team Software Process, el TSP extiende y refina los métodos de CMM y PSP sobre desarrollo y mantenimiento de equipos, y llegar a lo que se le llama un equipo auto dirigido.

**PROCESO**

La entrada de PSP son los requerimientos; el documento de requerimientos es completado y entregado al ingeniero.

**PSP0 tiene 3 fases:**

**PSP0 y PSP0.1** Introducen la disciplina y la medición al proceso

Planeación, desarrollo (diseño, codificación, pruebas) y un post mortem. Se establece una base del proceso normal de medición: tiempo tomado programando, fallos inyectados/removidos, tamaño de un programa. En un post mortem el ingeniero asegura que todos los datos del proyecto hayan sido registrados y analizados correctamente. **PSP0.1** agrega un estándar de código, una medida de tamaño y el desarrollo de un plan de mejora personal PIP. En el PIP el ingeniero registra ideas para mejorar su propio proceso.

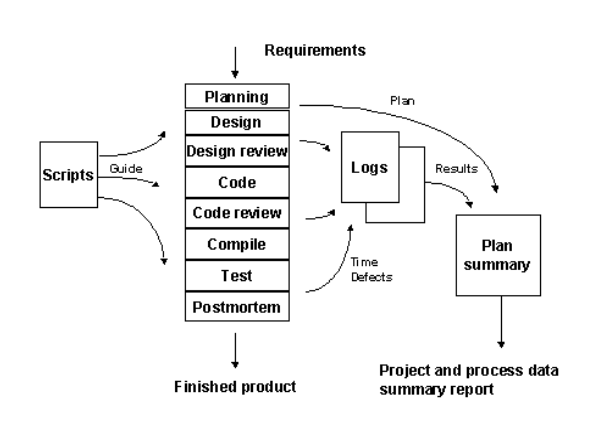
**12**

**PSP1, PSP1.1** (Introduce estimación y planeación)

Teniendo como base los datos recolectados en PSP0 y PSP0.1, el ingeniero estima el tamaño que tendrá el nuevo programa y prepara un reporte de pruebas (PSP1). Los datos recolectados para proyectos previos se usan para estimar el tiempo total. Cada proyecto nuevo registrará el tiempo gastado actualmente. Esta información es usada para tareas de agendamiento, planeación y estimación (PSP1.1).

**PSP2, PSP2.1 (Introduce manejo de calidad y diseño)**

PSP2 agrega dos fases nuevas: revisión de diseño y de código. Se enfoca en la prevención de defectos y su remoción. Los ingenieros aprenden a evaluar y mejorar su proceso midiendo la extensión de sus tareas y la cantidad de defectos inyectados y removidos en cada fase de desarrollo. Los ingenieros construyen y usan listas de chequeo para diseño y revisión de código.



**13**

**TSP**

**HISTORIA DEL TSP**

la version inicial del tsp fue desarrollada por watts humphrey en 1996, y el primer reporte tecnico para tsp fue publicado en el año 2000, patrocinado por el departamento de defensa de los estados unidos. el libro de watts humphrey llamado introduction to the team software process.

**QUE ES TSP**

es una metodologia para dirigir el trabajo de mejora y desarrollo de software ademas de establecer un entorno donde el trabajo efectivo de equipo sea normal y natural. conjunto de procesos estructurados que indican que hacer en cada fase del desarrollo del proyecto y muestra como conectar cada fase para construir un producto completo.

**OBJETIVOS**

* maximizar calidad software, minimizar costo.
* integrar equipos independientes de alto rendimiento que planeen y registren su trabajo, establezcan metas, y sean dueños de sus procesos y planes.
* mostrar a los gerentes como monitorear y motivar sus equipos de trabajo y como ayudarlos a alcanzar su maxima productividad.
* acelerar la mejora continua de procesos.
* proveer de unaguia para el mejoramiento en organizaciones maduras.

**14**

**Entornos**

Cmm administracion

tsp equipo ingenieros

psp ingeniero

**Fases del ciclo de vida**

* implementacion
* lanzamiento
* estrategia
* planeamiento
* requerimientos
* diseño
* pruebas
* postmorten

**RELACION PSP-TSP**

* ambos procesos pueden usarse juntos
* psp y el tsp son aplicables tanto a pequeña como a gran escala
* equipos sencillos, 5-15 profesionales
* multi-equipos, muchas docenas de profesionales.

**VENTAJAS**

* mejora la productividad de las personas.
* mejora en los habitos de programacion.
* se puede lograr una deteccion temprana de defectos y riesgos lo que deriva en una disminucion de los defectos.
* una mejora en la calidad.
* una reduccion en el ciclo de vida

**15**

**DESVENTAJAS**

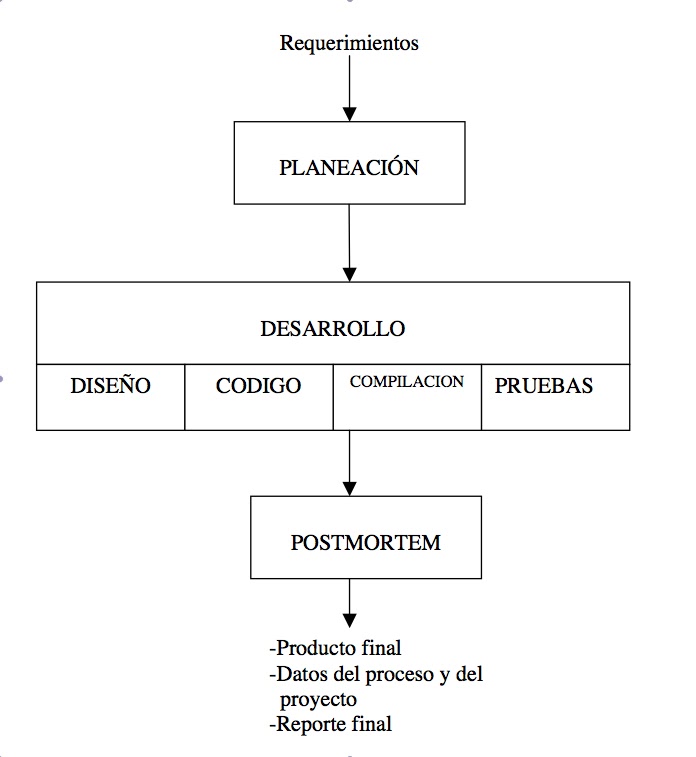
* Es necesario de cada uno de los miembros tiene que tener el compromiso y la disciplina de seguir el plan.
* Debe de llenar toda la documentacion requerida que incluye sus registros, planificacion, las plantillas o formularios.
* Se debe de contar con un buen cojunto de metricas y parametros de calidad, lo cual, para algunas organizaciones, puede ser dificil de definir.

**HERRAMIENTAS**

el introductory team software process(tsp) es una version academica-baja del tsp el cual guia graduantes y a estudiantes avanzados aplicando los principios y practica del tsp

\*scrum es un marco de trabajo para la gestion y desarrollo de software basada en procesos iterativo e incremental utilizado comunmente en entornos basado en el desarrollo agil de software.

\*aunque scrum estaba enfocado ala gestion de procesos de desarrollo de software, puede ser utilizando en equipos de mantenimiento de software, o en una aproximacion de gestion de programas.



**16**

**METODOLOGÍAS**

**Flujos de trabajo de apoyo**

**Gestión de Configuración:** Es la manera de controlar lo que desarrollan los miembros del equipo y algunos problemas.

**Gestión de proyectos:** Balancea los objetivos contrarios, maneja riesgos y produce SW que satisface al cliente y al usuario.

**Administración del ambiente:** Ambiente y herramientas de desarrollo que harán posible llevar a cabo el proyecto.

**Herramientas**

**Requisito Pro:** Es una herramienta que se apoya en el trabajo de equipo, basado en los requerimientos. Se integra a Microsoft Word para capturar documentos de requisitos.

**Racional Rose:** Es una herramienta de RATIONAL SOFTWARE CORPORATION con soporte UML, está orientado a la Ingeniería de Software (ISW), usado para el análisis, modelado, diseño y construcción del objeto orientado.

**Clear Case:** Es una herramienta para el conflicto entre versiones, administra los artefactos para crear modelos, codificación y pruebas. Aumenta la administración del espacio de trabajo incluyendo vistas dinámicas.

Definición de la metodología de desarrollo

Metodologías

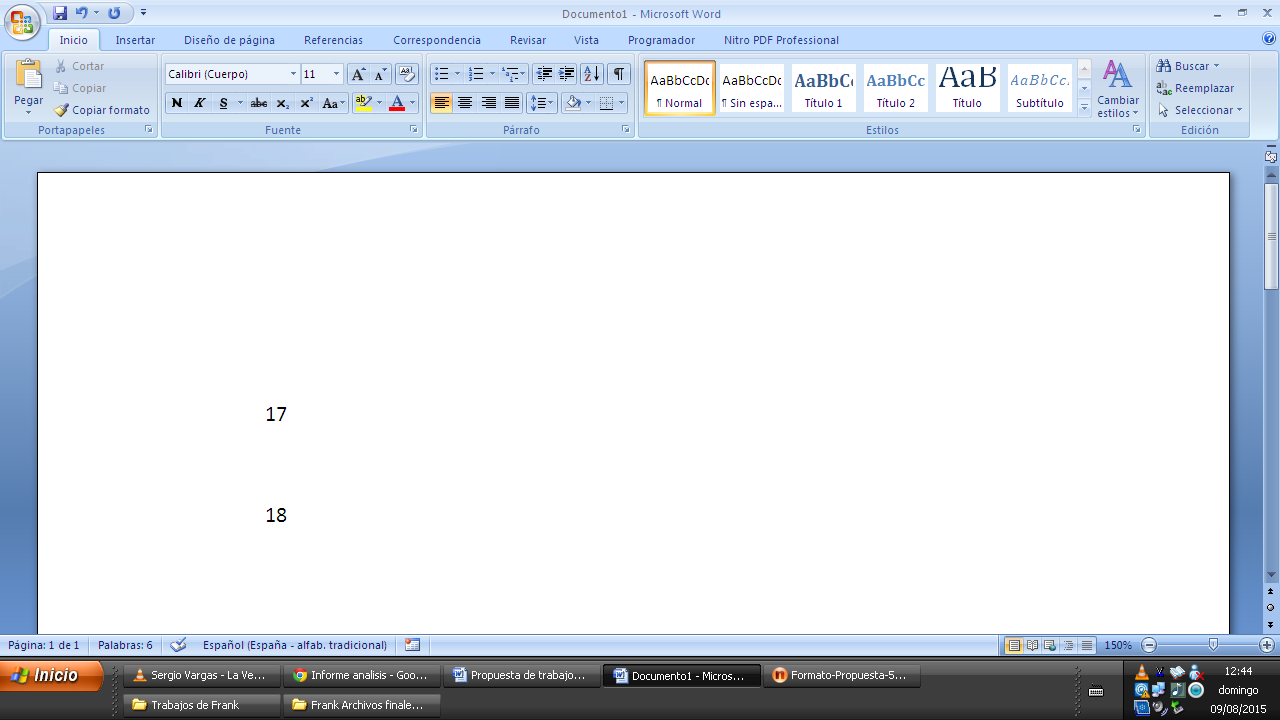
RUP

Este método se basa principalmente en cuatro valores o principios bien definidos:

1. **Simplicidad:** la simplicidad consiste en desarrollo solo el sistema que realmente se necesita. Implica resolver en cada momento solo las necesidades actuales.
2. **Feedback:** una metodología basada en el desarrollo iterativo de pequeñas partes, con entregas y pruebas frecuentes y continuas, proporciona un flujo de retroinformación valioso para detectar los problemas o desviaciones.
3. **Decisión:**

* Implica saber tomar decisiones difíciles.
* Reparar un error cuando se detecta.
* Mejorar el código siempre que tras el feedback y las sucesivas iteraciones.

XP



Como ustedes saben, Scrum es una metodología ágil para la gestión del desarrollo de software, que está basada en un proceso iterativo e incremental. Debido a la importancia de la arquitectura de software, es primordial definir claramente su papel en Scrum. Es por esto que en el presente artículo se describe una propuesta del papel de la arquitectura de software en el ciclo de desarrollo de Scrum y de las responsabilidades del arquitecto de software en esta metodología.

En los proyectos basados en Scrum se consideran tres roles:

* Dueño del producto (Product Owner): Es quien determina las prioridades de los entregables.
* Maestro de Scrum (Scrum Master): Administra y facilita la ejecución del proceso.
* Equipo de Trabajo (Stakeholders): Trabajan en conjunto para entregar resultados en cada sprint.

Definición de la metodología de desarrollo

Metodologías

SCRUMMMMMMMMMMM

El desarrollo en espiral es un modelo de ciclo de vida del software definido por primera vez por Barry Bohema en 1986,1 utilizado generalmente en la Ingeniería de software. Las actividades de este modelo se conforman en una espiral, en la que cada bucle o iteración representa un conjunto de actividades. Las actividades no están fijadas a ninguna prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior.

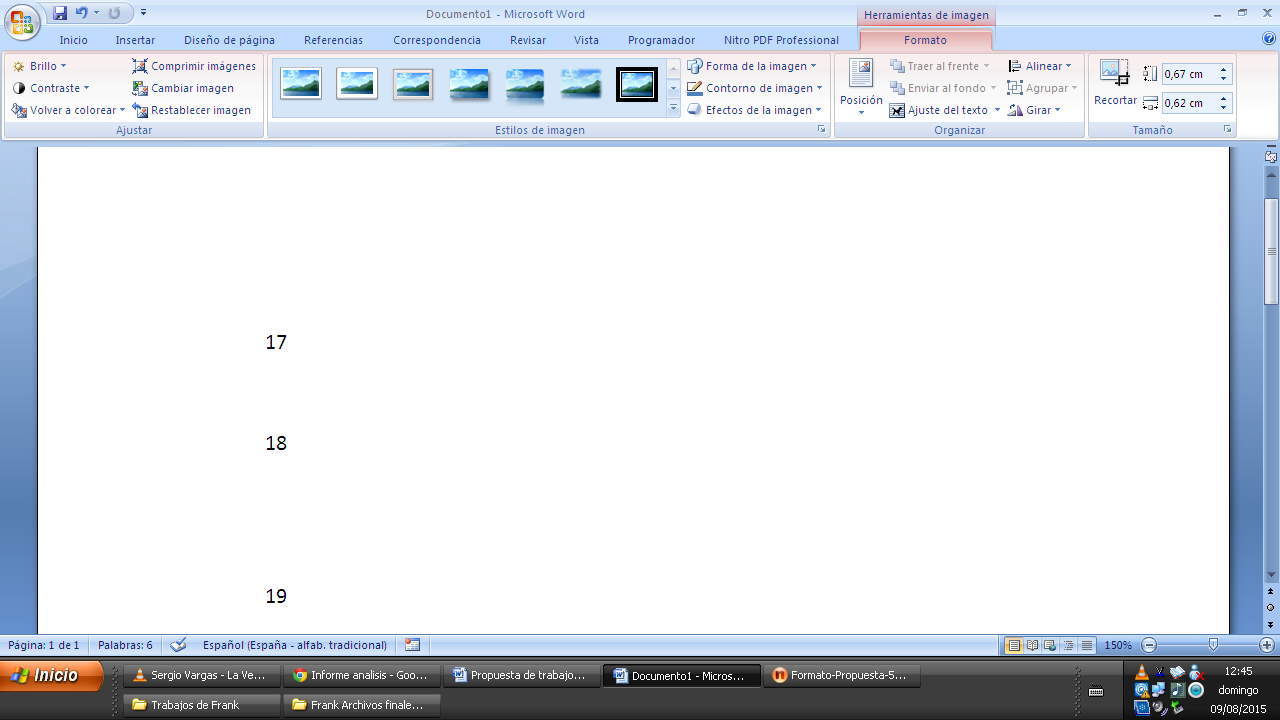
ESPIRAL

El modelo en cascada, algunas veces llamado el ciclo de vida clásico, sugiere un enfoque sistemático, secuencial hacia el desarrollo del software, que se inicia con la especificación de requerimientos del cliente y que continúa con la planeación, el modelado, la construcción y el despliegue para culminar en el soporte del software terminado.

Esta modelo es aplicable en donde existen ocasiones en que los requisitos de un problema se entienden de una manera razonable y deben estar bien definidos, también cuando el trabajo fluye desde la comunicación a través del despliegue de una manera casi lineal, esta situación se encuentra a veces cuando es necesario hacer adaptaciones o mejorías bien definidas a un sistema existente.

El modelo en cascada es el paradigma más antiguo para la ingeniería del software. Sin embargo, en las décadas pasadas, las críticas a este modelo de proceso han ocasionado que aun sus más fervientes practicantes hayan cuestionado su eficacia.

CASCADA O LINEAL



Los métodos agiles de desarrollo dependen de un enfoque iterativo para la especificación, desarrollo u entrega del software y principalmente fueron diseñando para apoyar al desarrollo de aplicaciones de negocios donde los requerimientos del sistema normalmente cambiaban rápidamente durante el proceso de desarrollo, están pensados para entregar software funcional de forma rápida a los clientes, quienes pueden que se incluya iteraciones posteriores del sistema o nuevos requerimientos o cambios en el mismo.

Los métodos agiles enfatizan las comunicaciones cara a cara en vez de la documentación, por esta razón es que generalmente los métodos agiles son criticados y tratados como “indisciplinados” por la falta de documentación técnica.

Definición de la metodología de desarrollo

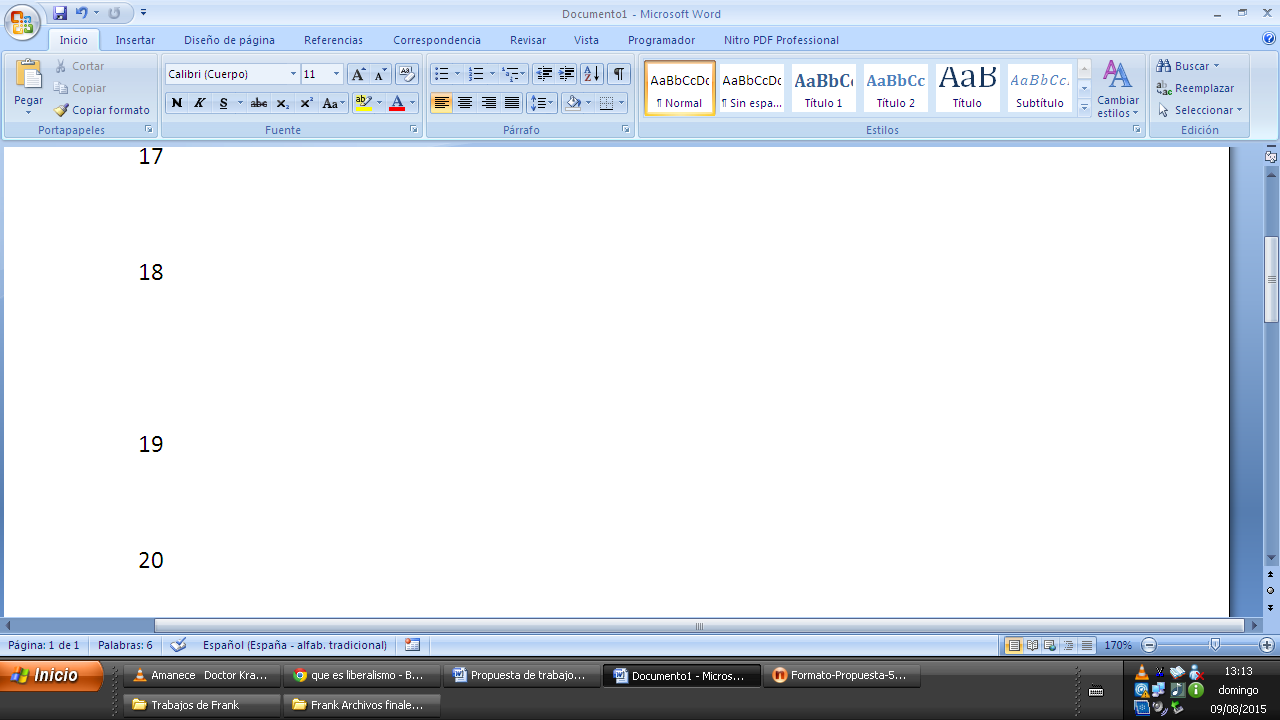
Metodologías

PROTOTIPOS

Es útil en situaciones donde los requerimientos se definen pobremente y/o cuando se necesita velocidad, para esto se requiere una administración efectiva para asegurar que las interacciones en la elaboración de prototipos no continuarán indefinidamente. Es importante contar con herramientas como lenguajes de software de cuarta generación y generadores de pantalla

El desarrollo rápido de aplicaciones o RAD (Rapid Application Development) es un proceso de desarrollo de software, el método comprende el desarrollo iterativo, la construcción de prototipos y el uso de utilidades CASE.

CRYSTAL



El software evoluciona con el tiempo. Los requisitos del usuario y del producto suelen cambiar conforme se desarrolla el mismo. Las fechas de mercado y la competencia hacen que no sea posible esperar a poner en el mercado un producto absolutamente completo, por lo que se aconsejable introducir una versión funcional limitada de alguna forma para aliviar las presiones competitivas. En esas u otras situaciones similares los desarrolladores necesitan modelos de progreso que estén diseñados para acomodarse a una evolución temporal o progresiva, donde los requisitos centrales son conocidos de antemano, aunque no estén bien definidos a nivel detalle. En el modelo cascada y cascada realimentado no se tiene demasiado en cuenta la naturaleza evolutiva del software, se plantea como estático, con requisitos bien conocidos y definidos desde el inicio. Los evolutivos son modelos iterativos, permiten desarrollar versiones cada vez más completas y complejas, hasta llegar al objetivo final deseado; incluso evolucionar más allá, durante la fase de operación.

EVOLUTIVO

RAD

Las iteraciones se pueden entender como mini proyectos: en todas las iteraciones se repite un proceso de trabajo similar (de ahí el nombre “iterativo”) para proporcionar un resultado completo sobre producto final, de manera que el cliente pueda obtener los beneficios del proyecto de forma incremental. Para ello, cada requisito se debe completar en una única iteración: el equipo debe realizar todas las tareas necesarias para completarlo (incluyendo pruebas y documentación) y que esté preparado para ser entregado al cliente con el mínimo esfuerzo necesario. De esta manera no se deja para el final del proyecto ninguna actividad arriesgada relacionada con la entrega de requisitos.

En cada iteración el equipo evoluciona el producto (hace una entrega incremental) a partir de los resultados completados en las iteraciones anteriores, añadiendo nuevos objetivos/requisitos o mejorando los que ya fueron completados. Un aspecto fundamental para guiar el desarrollo iterativo e incremental es la priorización de los objetivos/requisitos en función del valor que aportan al cliente.

Definición de la metodología de desarrollo

Metodologías

INCREMENTAL E ITERATIVO

**CASCADA POR SUBPROYECTO**

Se entiende como una variación sobre el ciclo de vida en Cascada del software, denominada Cascada con Subproyecto, porque permite la ejecución de algunas de las tareas de la cascada en paralelo.

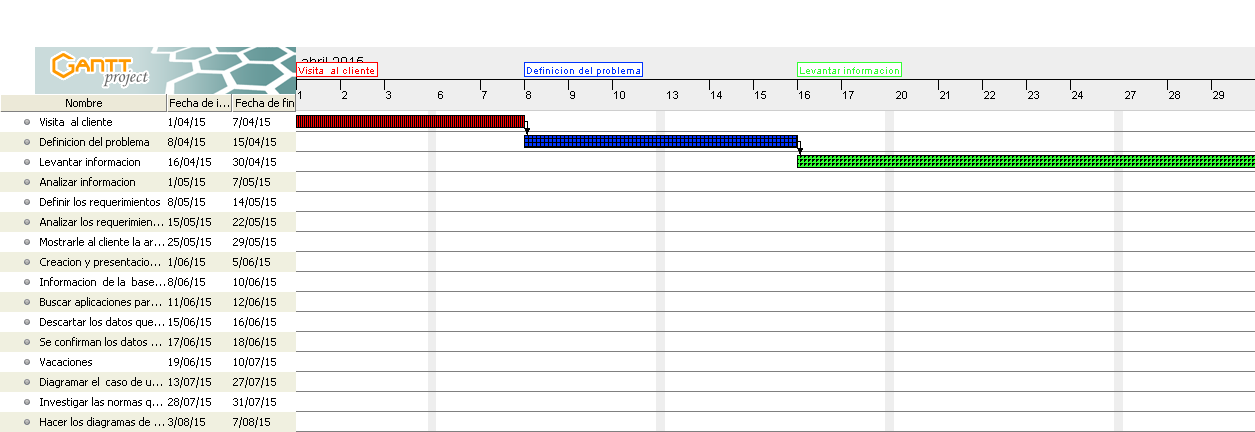
Un ejemplo de aplicación de esta metodología podría ser el desarrollo de un sistema de información para una empresa, en donde deben estar involucradas todas las áreas de la misma porque siempre están compartiendo información. Para ello se pueden ir adelantando de forma paralela las etapas del ciclo de vida como análisis, diseño, desarrollo y pruebas de forma independiente para cada área de la organización y hacia el final se realiza la integración de los resultados de cada subproyecto.

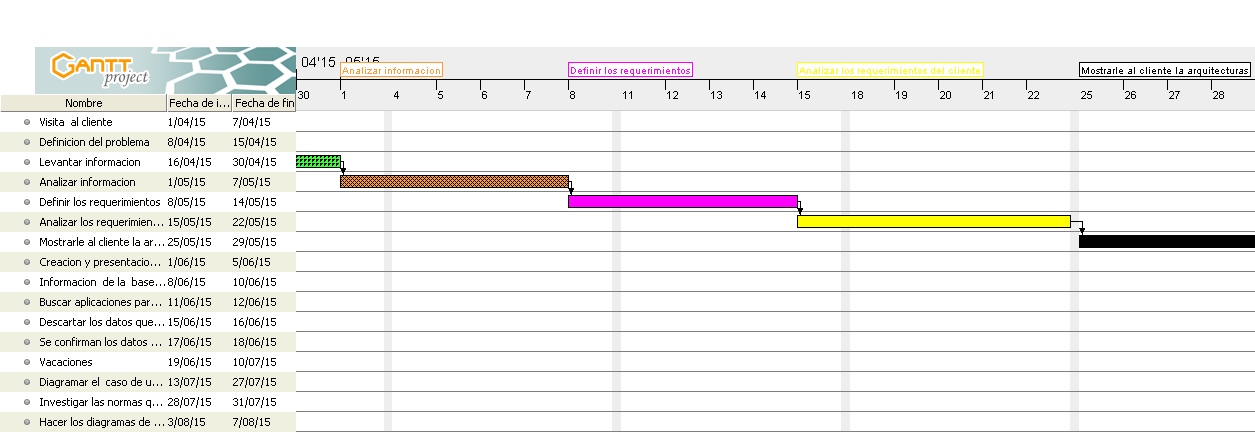
**20**

**Diagrama de Gantt**

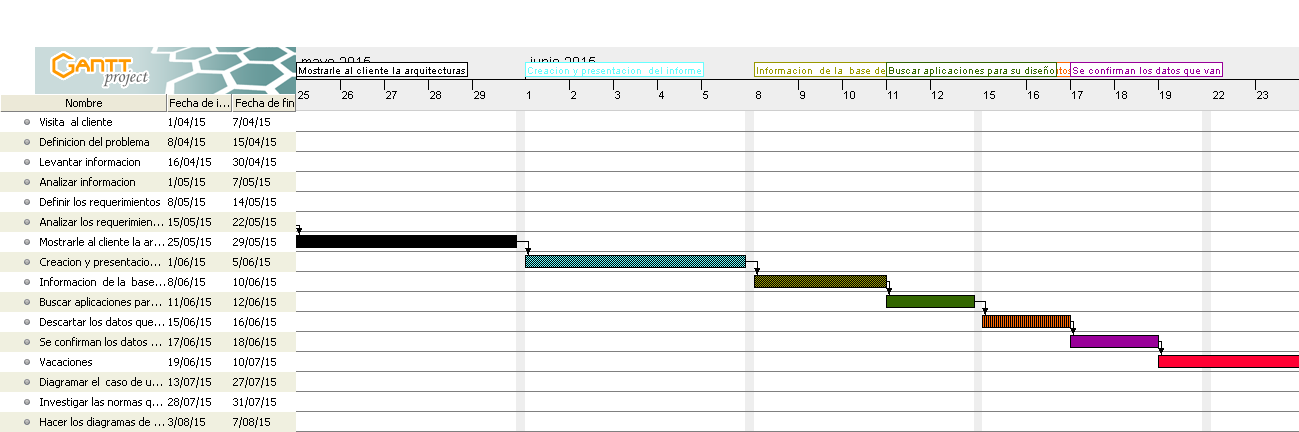
**Método usado para hacer nuestro proyecto es:**

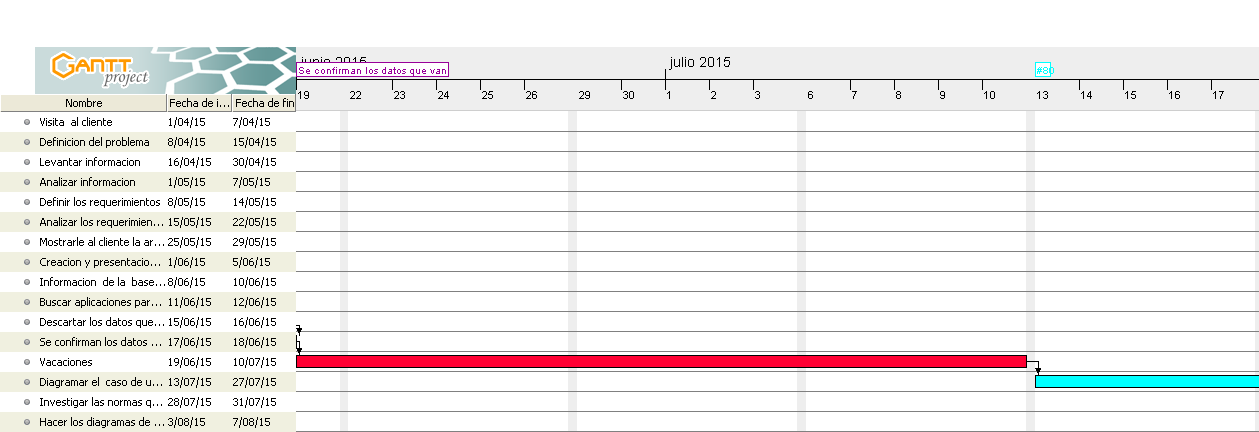
El método Cascada o Lineal Observado en el Diagrama de Gantt del proyecto





**21**



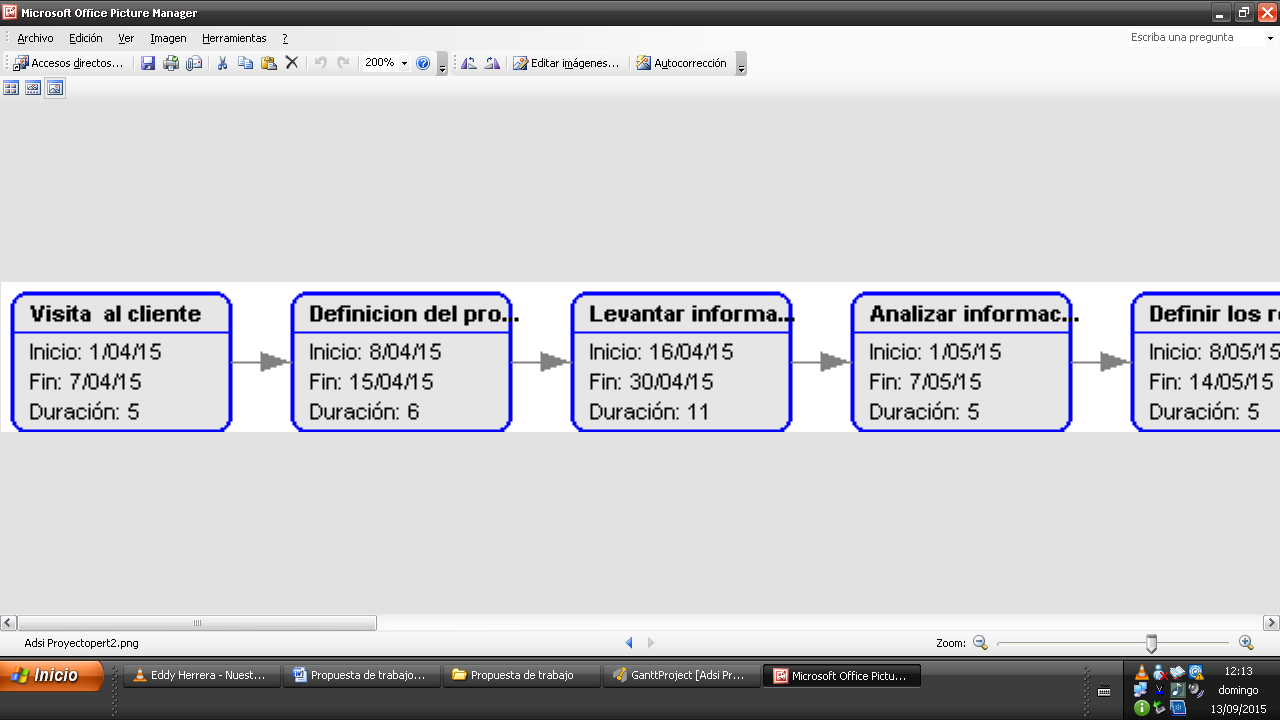


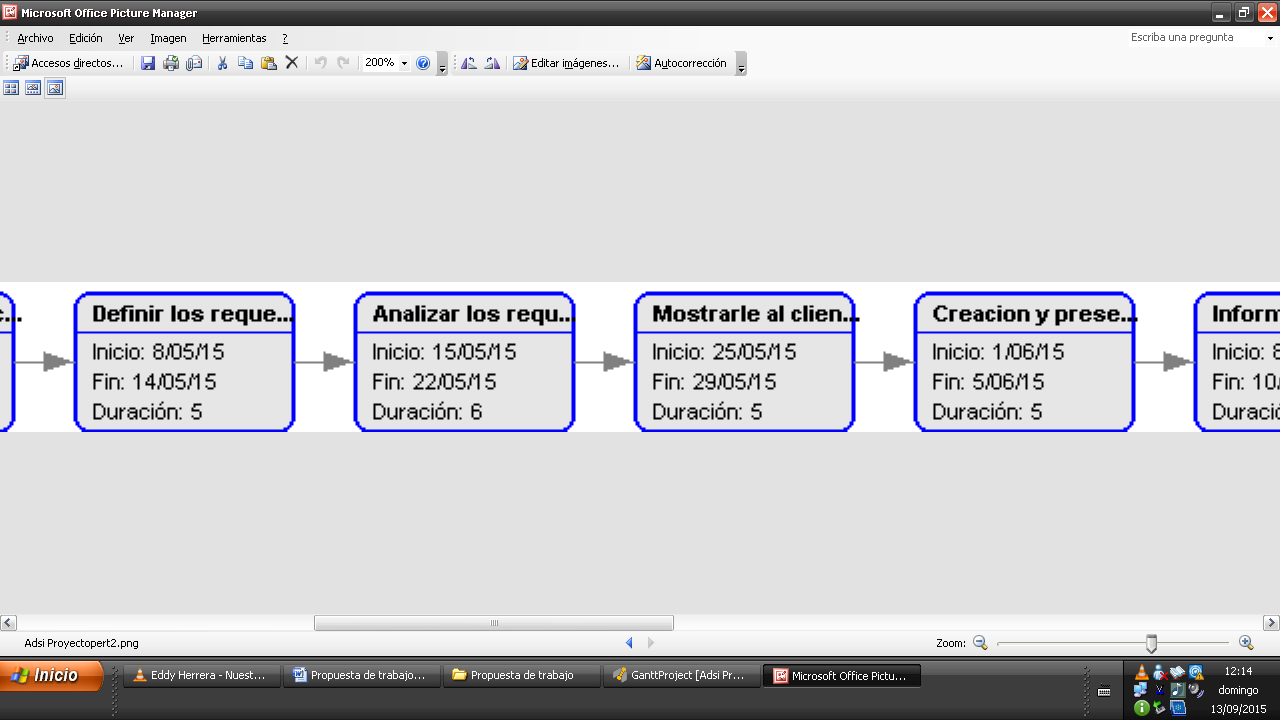
**22**

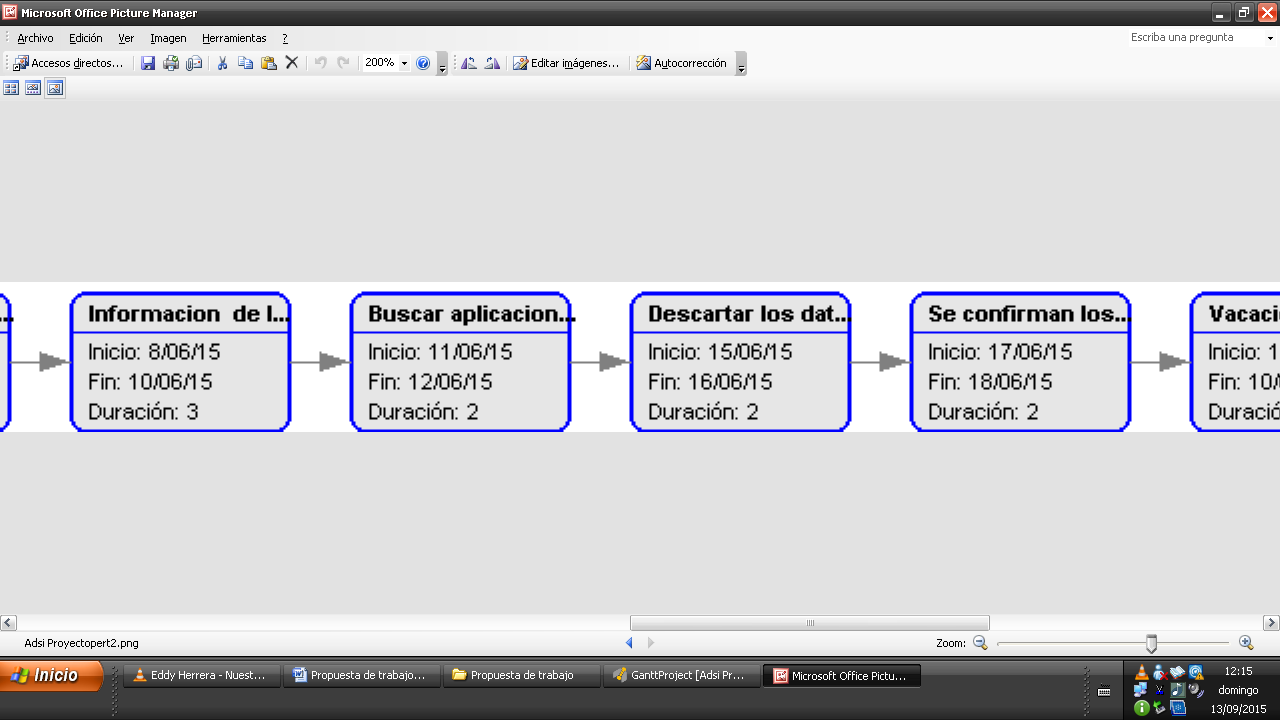
**Diagrama de Pert**

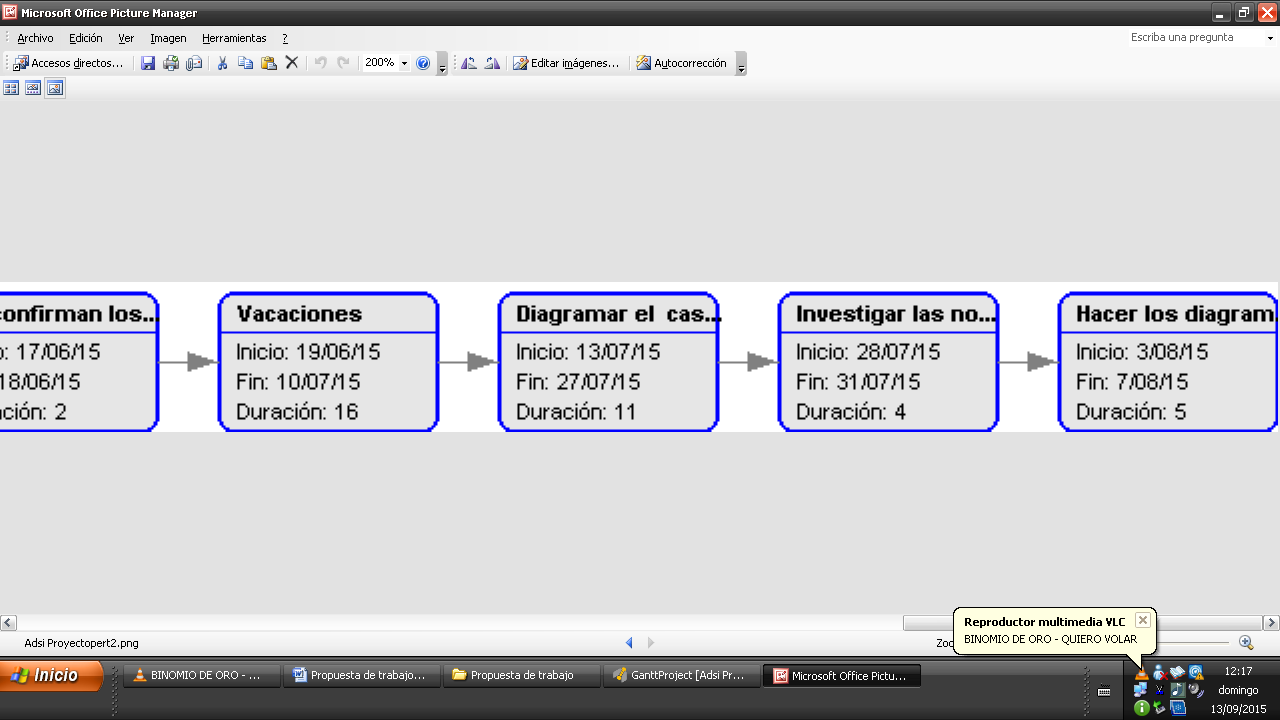
**Método usado para hacer nuestro proyecto es:**

El método Cascada o Lineal Observado en el Diagrama de Pert del proyecto

****

****

****

****

**23**

**BIBLIOGRAFÍA**

<https://sisteminformacii.wikispaces.com/METODOLOG%C3%8DA+DE+KENDALL+%26+KENDALL>

<http://administrandoproyectos.blogspot.com/2011/01/psp-personal-software-process.html>

<http://es.slideshare.net/dlpoma/team-software-process-tsp>+

**24**