Resumen Teorico Ingenieria De Software 1

Teoria1 – Elicitacion Requerimientos :

.El sw no sigue un envejecimiento clasico, ya que este al inicio posee una alta tasa de fallos la cual se va solucionando con actualizaciones, pero por cada cambio se incrementa esta tasa de fallos por efectos colaterales generando un nuevo pico de tasa de fallos y asi repetidamente, obviamente con el paso del tiempo esta curva tiende a ir en aumento. El problema no esta en el tiempo de operación (como sucede con el hw), sino en los cambios.

.El sw tiene dos formas como producto:

Generico: Sistema aislado producido por organizaciones desarroladoras de sw y que se venden en un mercado abierto

Personalizados: Sistema requerido por un cliente particular.

Cada vez mas sistemas son construidos como un producto generico como base que luego es adaptado a los requerimientos del cliente.

Libertades del sw libre:

1. Libertad para ejecutar el programa en cualquier sitio.

2. Libertad para estudiarlo y adaptarlo a nuestras necesidades.

3.Libertad de redistribucion, de modo que nos permite colaborar con vecinos y amigos.

4.Libertad para mejorar el programa y publicar las mejoras.

.La ingeniera de sw es la disclipna de la ingenieria que comprende todos los aspectos de la produccion de sw desde las etapas iniciales de la especificacion del sistema incluyendo la evolucion de este, luego que se comienza a ejecutar. Esta “hace que las cosas funcionen” y no solo comprende los procesos tecnicos del desarrollo de sw, sino tambien se realizan actividades como la gestion de proyectos y el desarrollo de herramientas, metodos y teorias de apoyo a la produccion de sw.

La IEEE la define como el uso de metodos sistematicos, disciplinados y cuantificables para el desarrollo, operación y mantenimiento de sw.

Un ingeniero de sw debe dominar los aspectos tecnicos, aprender habilidades requeridas para entender el problema, diseñar soluciones, etc. Pero ademas, los aspectos humanos es lo que hacen un ingeniero efectivo. Tener un sentido de responsabilidad individual, tener conciencia grupal, atencion al detalle entre otros. Ademas de la responsabilidad profesional y etica que debe tener uno, ya que se desarrolla en un marco economico, social y legal, entre ellas: confidencialidad, competencia, derechos de la propiedad intelectual, uso inapropiado de las computadoras.

Tecnicas de comunicación:

.Al iniciar un proyecto debemos saber lo que el usuario quiere, como lo quiere, cuando y porque. Para esto es importante la comunicación.

.Al hablar de las necesidades del cliente, se habla de Requerimientos.

.Un requerimiento es una caracteristica del sistema o descripcion de algo que el sistema es capaz de hacer con el objeto de satisfacer el proposito del sistema.

Fuentes de requerimientos: Documentacion, stakeholders, especificaciones de sistemas similares

Stakeholder: referencia a cualquier persona o grupo que se vera afectado por el sistema, directa o indirectamente, de los stakeholders se pueden obtener diferentes visiones.

Puntos de vista: existen tres tipos genericos:

Punto de vista de los interactuadores: representan a las personas u otros sistema que interactuan directamente con el sistema.

Punto de vista indirecto: representan a los stakeholders que no utilizan el sistema ellos mismos pero influyen en los requerimientos de algun modo.

Punto de vista del dominio: representan las caracterisiticas y restricciones del dominio que influyen en los requerimientos del sistema.

Elicitacion de requerimientos:

.Es el proceso de adquirir todo el conocimiento relevante necesario para producir un modelo de los requerimientos de un dominio de problema. Es una actividad principalmente de carácter social, mucho mas que tecnologico. Los problemas que se plantean son por tanto de naturaleza psicologica y social, mas que tecnicos.

Como dificultades en esta etapa tenemos: problemas de comunicación, limitaciones cognitivas del desarrollador (no conocer el dominio del problema), dificultades por la conducta humana y dificultades tecnicas (complejidad del dominio o requisitos).

Tecnicas de elicitacion:

Metodos discretos: son menos pertubadores que otras formas de averiguar los requerimientos pero estos se consideran insuficientes para recopilar info cuando se utilizan por si solos. Utilizar varios metodos para acercarse a la organización es una practica inteligente.(Muestreo de documentacion, formularios y datos existentes, investigacion y visitas al sitio, observacion del ambiente de trabajo).

Metodos interactivos: estos metodos tienen muchas cosas en comun, la base es hablar con las personas en la organización y escuchar para comprender. Cada uno cuenta con su propio proceso establecido.

Planeacion conjunta de requerimientos: Proceso mediante el cual se conducen reuniones de grupo altamente estructurados con el proposito de analizar problemas y definir requerimientos.

Brainstorming: Tecnica para gnerar ideas al alentar a los participantes para que ofrezcan tantas ideas como sea posible en un corto plazo sin ningun analisis hasta que se hayan agotado las ideas.

Cuestionarios: documento que permite recabar informacion y opiniones de los encuestados. Este puede poseer preguntas abiertas o cerradas.

Entrevistas: tecnica de exploracion mediante la cual el analista de sistemas recolecta info de las personas a traves de la interaccion cara a cara. Existen estructuradas (cerradas), no estructuradas (abiertas) en estas se utilizan tres tipos de preguntas, que son abiertas, cerradas y de sondeo. Dado este tipo de preguntas la entrevista se puede organizar de tres formas: Piramidal (inductivo) va de preguntas cerradas a abiertas, Embudo (deductivo) va de preguntas abiertas a cerradas y Diamante (como combinacion de las anteriores) empieza con preguntas cerradas pasa por un tramo de preguntas abiertas para finalmente terminar con preguntas cerradas. Leer recomendaciones de Kendall y Whitten (puede ser interesante)

Reflexionar beneficios y desventajas que tiene cada una, es un buen trabajo para entender cada una ademas de solamente leerlas una y otra vez, me parece mejor entender cada una que sabermelas de memoria.

Teoria 2 – Requerimientos:

Tipos:

Requerimientos Funcionales: Describen una interaccion entre el sistema y su ambiente. Describen lo que el sistema debe hacer, o incluso como NO debe comportarse.

Requerimientos no Funcionales: describen una restriccion sobre el sistema que limita nuestras elecciones en la construccion de una solucion al problema. Existen tres: del producto (especifican su comportamiento), organizacionales (derivan de las politicas y procedimientos existentes en la organización del cliente y en la del desarrollador) y externos (legales, privacidad, seguridad, eticos).

Ingenieria de requerimientos: es el proceso por el cual se transforman los requerimientos declarados por los clientes, ya sean hablados o escritos, a especificaciones precisas, no ambiguas, consistentes y completas del comportamiento del sistema, incluyendo funciones, interfaces, rendimiento y limitaciones. Este proceso utiliza una combinacion de metodos, herramientas y actores, cuyo producto es un modelo del cual se genera un documento de requerimientos.

Esto se puede ver como un proceso secuencial e iterativo en el cual las actividades se ven entrelazadas. (especificaciones, adquisiciones y validaciones).

Estudio de viabilidad: Es la primera etapa de la ingenieria, esta se da principalmente para sistemas nuevos. A partir de una descripcion resumida del sistema se elabora un informe que recomienda la conveniencia o no de realiza el proceso de desarrollo. Responde a: el sistema contribuye a los objetivos generales de la organización? Puede implementar la tecnologia actual? Puede implementar restricciones de costo y tiempo? Puede integrarse a otros que existen en la organización? Una vez recopilada toda la info se deberia hablar con fuentes de infor para responder nuevas preguntas y redactar un nuevo informe donde deberia hacerse una recomendación sobre si se debe continuar o no el desarrollo.

Especificacion de requerimientos:

Propiedades de los requerimientos: necesario, consico, completo, consistente, no ambiguo, verificable.

Esta etapa tiene como objetivos:

.Permitir que los desarrolladores expliquen como han entendido lo que el cliente pretende del sistema.

.Indicar a los diseñadores que funcionalidad y caracteristicas va a tener el sistema resultante.

.Indicar al equipo de pruebas que demostraciones llevar a cabo para convencer al cliente.

En esta etapa se utilza documentacion para los requerimientos (de definicion y especificacion).

Aspectos basicos de una especificacion de requerimientos: funcionalidad, interfaces externas(interaccion), rendimiento, atributos, restricciones de diseño.

Un documento de requerimientos es utilizado por: clientes del sistema, administradores, ingenieros del sistema, ingenieros de prueba del sistema, ingenieros de mantenimiento del sistema. (Obviamente cada uno con un proposito distinto, pensar que hace cada uno con este, es intuitivo (pag 23)).

Validacion de Requerimientos:

.Es el proceso de certificar la correcion del modelo de requerimientos contra las intenciones del usuario. Este trata de mostrar que los requerimientos definidos son los que estipula el sistema. En este se describe el ambiente en el que debe operar el sistema.

Definiciones:

Validacion: al final del desarrollo evaluar el sw para asegurar que el sw cumple los requerimientos (hacer el sw correcto)

Verificacion: el sw cumple los requerimientos correctamente. (hacer el sw correctamente)

.La validacion solo se puede hacer con la activa participacion del usuario.

.Validar en la fase de especificacion de requerimientos puede ayudar a evitar costosas correciones despues del desarrollo.

Verificaciones que comprenden: de validez (para todos los usuarios), de consistencia (sin contradicciones), de completitud (todos los requerimientos), de realismo (se pueden implementar), verificabilidad (se puede diseñar conjunto de pruebas).

Tecnicas de validacion: estas pueden ser manuales o automatizadas.

Revisiones de requerimientos (formales o informales), las informales se tratan con tantos stakeholders como sea posible y en las formales el equipo de desarrollo debe conducoir al cliente, explicandole las implicaciones de cada requerimiento. Antes de una revision formal, es conveniente realizar una informal.

Tecnicas de especificacion de requerimientos de software:

Estaticas: se describe el sistema a traves de las entidades u objetos, sus atributos y sus relaciones con otros sin cambios en el tiempo. Cuando el tiempo no es un factor mayor en la operación del sistema, es una descripcion util y adecuada. Ej: relaciones de recurrencia, expresiones regularesm abtracciones de datos.

Dinamicas: se considera un sistema en funcion de los cambios que ocurren a lo largo del tiempo. Se considera que el sistema esta en un estado particular hasta que un estimulo lo obliga a cambiar su estado.

Ej: tablas de decision, dte, tablas de transicion de estados, Redes de Petri, etc.

Historias de Usuario: Es una descripcion corta y simple de un requerimiento de un sistema, que se escribe en lenguaje comun del usuario y desde su perspectiva. Son utilizadas en las metodologias de desarrollos agiles (XP,SCRUM) para la especificacion de requerimientos.

Esta debe ser limitada, es una forma rapida de administrar los requisitos, responden rapidamente a los requisitos cambiantes, deben tener la posibilidad de discutirlas con los clientes, como estimacion de tiempo cada una se debe situar entre 10 horas y un par de semanas, estimaciones mayores a dos semanas son indicactivos de que la historias es muy compleja y esta debe ser dividida.

Esquema: como ROL quiero ALGO para BENEFICIO.

Caracterisiticas: independientes unas de otras, negociables, valoradas por los clientes o usuarios, estimables, pequeñas, verificables.

Criterio de aceptacion: es el criterio por el cual se define si una historia de usuario fue desarrollada según la expectativa del Owner y si se puede dar como hecha. Estos deben ser definidos durante la etapa inicial de la codificacion, acompañan a la HU, porque complementan la HU y ayudan al equipo de desarrollo a entender mejor como se espera que el producto se comporte. Estas son el resultado de las conversaciones del cliente con el desarrollador. Si tiene mas de 4, debe evaluarse subdividir la historia.

Plantilla: id, titulo, reglas de negocio, criterios de aceptacion, escenarios.

Epica: conjunto de HU que se agrupan por algun denominador comun. Esta representa un objetivo alcanzable que nace de la necesidad del cliente, la epica NO es la funcionalidad. Es una oracion que cumple con el esquema de HU pero se puede desentrañar para hacer mas de 1 HU debido a su complejidad. Estas suelen abarcar varios equipos de desarrollo, los cliente son quienes determinan si eliminar o añadir historias a una epica, sirve para estructurar los objetivos ademsaq sirven para dar flexibilidad y agilidad al proyecto.

Teoria 3 – Tecnicas de especificacion de requerimientos Casos de uso:

Casos de uso: proceso de modelado de las funcionalidades del sistema en termino de los eventos que interactuan entre el usuario y el sistema.

Es una herramienta para capturar requerimientos funcionales, es un medio de comunicación con los usuarios, permite estimar el alcance del proyecto y el esfuerzo a realizar, entre otros beneficios.

Caso de uso: representa un objetivo (funcionalidad) individual del sistema y describe la secuencia de actividades y de interacciones para alcanzarlo. Para que sea considerado un requerimiento debe estar acompañado de su respectivo escenario.

Actor: inicia una actividad (CU) en el sistema. Representa un papel desempeñado por un usuario que interactua (rol). Puede ser una persona, sistema externo o dispositivo externo que dispare un evento(sensor, reloj).

Asociacion: relacion entre un actor y un CU en el que interactuan entre si.

Extension: los CU extensiones son iniciados por un CU.

Uso o inclusion: reduce la redundancia entre dos o mas CU al combinar los pasos comunes de los CU.

Herencia: relacion entre actores donde un actor hereda las funcionalidades de uno o varios actores.

Proceso de modelado:

Identificar los actores: quien proporciona las entradas al sistema?

Identificar los CU para los requerimientos: cuales son las principales tareas del actor?

Construir el diagrama:

Realizar los escenarios

.Un CU debe representar una funcionalidad concreta.

Teoria 4 – DTE:

Maquinas de estado finito: describe al sistema como un conjunto de estados donde el sistema reacciona a ciertos eventos posibles (externos o internos).

{ F(si,cj) = sk } => Al estar en el estado si, la ocurrencia de la condicion cj hace que el sistema cambie al estado sk.

Definicion formal: un automata finito (AF) puede ser descrito como una 5-tupla (S,E,T,s,A) donde:

E es un alfabeto.

S un conjunto de estados.

T es la funcion de transicion.

S es el estado inicial.

A es un conjunto de estados de aceptacion o finales.

Evento: es un suceso significativo que debe tenerse en cuenta, que influye en el comportamiento y evolucion del sistema. Tiene lugar en un punto del tiempo y carece de duracion respecto a la granularidad temporal del sistema.

Transicion: se producen como consecuencia de eventos. Pueden o no tener un procesamiento asociado.

Transicion = Evento [condicion] / accion

Construccion de un DTE

1. Identificar los estados

2. Si hay un estado complejo se puede explotar

3. Identificar el estado inicial

4. Desde el estado inicial, se identifican los cambios de estado con flechas

5. Se analizan las condiciones y las acciones para pasar de un estado a otro

6. Se verifica la consistencia: Se han definido todos los estados, se pueden alcanzar todos, se puede salir de todos y en cada uno, el sistema responde a todas las condiciones posibles.

Teoria 5 – Redes de Petri:

.Las tareas concurrentes deben estar sincronizadas para permitir la comunicación entre ellas (pueden operar a distintas velocidades, deben prevenir la modificacion de datos compartidos o condiciones de bloqueo). Pueden realizarse varias tareas en paralelo, pero son ejecutados en un orden impredecible. Estas NO son secuenciales.

Eventos = Transiciones (T)

Estados = sitios (P)

Definicion formal: una estructura de Red de Petri es una 4-tupla

C =(P,T,I,O)

.P = lugares

.T = transiciones

.I = funcion de entrada I

.O = funcion de salida O

.El numero de token asignados a un sitio es ilimitado

.El conjunto de tokens asociado a cada estado sirve para manejar la coordinacion de eventos y estados. La ejecucion es controlada por el numero y distribucion de los tokens.

Sincronizacion: Para que varios procesos colaboren en la solucion de un problema es necesario que compartan info y recursos pera esto debe ser controlado para asegurar la integridad y correcta operación del sistema.

Casos particulares: Expresion de exclusion mutua, Productor – Consumidor, Condicion de bloqueo.

.Las RdP pueden ser aplicadas para la modelacion de sistemas de eventos discretos, las cuales ofrecen una forma de representacion grafica y matematica de los sistemas modelados. La formalidad matematica de las RdP proporcionan herramientas de analisis para analizar los posibles estados a los que el sistema modelado pudiera alcanzar.

Teoria 6 – Tablas de decision:

.Es una herramienta que permite presentar de forma concisa las reglas logicas que hay que utilizar para decidir acciones a ejecutar en funcion de las condiciones y la logica de decision de un problema especifico. Esta describe el sistema como un conjunto de:

.posibles CONDICIONES satisfechas por el sistema en un momento dado REGLAS para reaccionar ante los estimulos que ocurren cuando se reunen determinados conjuntos de condiciones y ACCIONES a ser tomadas como un resultado.

La tabla se contruye con condiciones y acciones simples. Las condiciones toman solo valores de V o F y hay 2 a la N reglas donde N es la cantidad de condicionales. (Recordar el 2 a la N)

Construccion de la tabla: Identifiar condiciones y acciones, tener en cuenta “condiciones opuestas” y “condiciones atomicas” y por ultimo construir las reglas.

.Especificaciones completas: determinan acciones (una o varias) para todas las reglas posibles

.Especificaciones redundantes: marcan para reglas que determinan las mismas condiciones acciones iguales.

.Especificaciones contradictorias: especifican para reglas que determinan las mismas condiciones acciones distintas.

Regla = condiciones + accion. Seria el conjunto de ambas.

Reduccion de complejidad (Redundancia): Combinar las reglas en donde sea evidente que una alternativa no represetna diferencia en el resultado. (Se utiliza un guion). Nos ahorramos columnas.

Entiendo por los ejemplos que no se puede tener una fila sin ninguna acciones, por lo cual si esto sucede se deberia “crear” una nueva accion para realizar en esa fila. PREGUNTAR.

Analisis estructurado: Permite lograr una representacion grafica que permite lograr una comprension mas profunda del sistema a construir y comunicar a los usuarios lo comprendido. La notificacion no especifica aspectos fisicos de implementacion. Esta hace enfasis en el procesamiento o la transformacion de datos conforme estos pasan por distintos procesos.

Diagrama de flujo de datos (DFD): Es una herramienta que permite visualizar un sistema como una red de procesos funcionales, conectados entre si por “conductos” y almacenamiento de datos. Representa la transformacion de entradas a salidas y es tambien llamado diagrama de burbujas. Es comunmente utilizada por sistemas operacionales en los cuales las funciones del sistema son de gran importancia y son mas complejas que los datos que este maneja.

Componentes del DFD: entidad externa (sistema que produce info para ser transformada por el sw o recibe info producida por el sw), proceso (es aplicado a los datos y los modifica), elemntos de datos (se representa como una flecha y es un conjunto de estos), almacen de datos (“fuente infinita de datos”).

Desarrollo de DFDs: Esta se debe visualizar desde una perspectiva jerarquica de arriba hacia abajo.

.Primero se redacta la lista de eventos de la organización para determinar los componentes.

.Luego se crea un diagrama de contexto con las entidades externas y los flujos de datos desde y hacia el sistema. Este muestra un panorama globla que muestra las entradas y salidas basicas.

.Se dibuja un diragama 0 (sig nivel), con procesos generales y los almacenes correspondientes. Es la ampliacion del diagrama de contexto, las entradas y salidas del diagrama de contexto permanecen, sin embargo, se amplia para incluir hasta 9 procesos (como max) y mostrar los almacenes de datos y nuevos flujos.

.Por ultimo se dibuja un diagrama hijo por cada uno de los procesos del diagrama 0. Las entradas y salidas del procesos padre permanecen, sin embargo, pueden aparecer nuevos almacenes de datos y nuevos flujos.

Teoria 7 – Modelos de Proceso:

Proceso de Sw: conjunto de actividades y resultados asociados que producen un producto de sw. Este posee cuatro actividades fundamentales que son:

.Especificacion del sw: proceso de comprender y definir que servicios se requieren del sistema, asi como la identificacion de las restricciones sobre las operación y desarrollo delo sistema. Tambien llamada, Ingenieria de Requerimientos.

.Desarrollo del sw: proceso de convertir una especificacion del sistema en un sistema ejecutable. Incluyr los procesos de diseño y programacion.

.Validacion del sw: se realiza para mostrar que un sistema cumple tanto con sus especificaciones como con las expectativas del cliente.

.Evolucion del sw: el mantenimiento es una actividad a tener en cuenta en el proceso de desarrollo de sw. Esto implica tambien cambios y mejoras.

Modelo de proceso de sw: es una representacion simplificada de un proceso de sw que presenta una vision de ese proceso. Pueden incluir actividades que son partes de los procesos y productos de sw, y el papel de las personas involucradas. Este abarca la vida del sistema desde la definicion de los requisitos hasta la finalizacion de su uso.

Los terminos modelo de proceso, paradigma de sw y ciclo de vida del sw son tres terminos equivalentes.

Tipos de modelos:

.Prescriptivos: prescriben un conjunto de elementos del proceso: actividades del marco de trabajo, acciones de la ing de sw, tareas, aseguramiento de la calidad y mecanismos de control. Cada modelo de proceso tambien prescribe un “flujo de trabajo”, es decir de que forma los elementos del proceso se interrelacionan entre si.

.Descriptivos: describen en la forma en que se realizan en la realidad. Ambos modelos deberian ser iguales.  
.Tradicionales: estan formados por un conjunto de fases o actividades en las que no se tienen en cuenta la naturaleza evolutiva del sw: clasico (lineal o en cascada), Modelo en V y basados en prototipos.

.Evolutivos: estos se adapatan a la evolucion que sufren los requisitos del sistema en funcion del tiempo: En esprial, evolutivo (estos dos tienen un desarrollo por fases) e incremental.

.Agiles: -

.Modelo en cascada: las etapas se representan cayendo en cascada, cada etapa de desarrollo se debe completar antes que comience la siguiente, es util para diagramar lo que se necesita hacer y su simplicidad hace que sea facil explicarlo a los clientes. Las dificultades que presenta, entre otras, son: no existen resultados concretos hasta que todo este terminado, el manejo de fallos se vuelve complejo ya que las mas triviales se generan al inicio del periodo de prueba y las mas graves al final, lo que las hace mas complejas de manejar, ademas de esta forma es dificil tener en cuenta cambios de parecer/ cambios en el ambiente.

.Modelo en V: demuestra como se relacionan las actividades de prueba con las de analisis y diseño, ademas sugiere que la prueba unitaria y de integracion tambien sea utilizada para verificar el diseño del programa. La vinculacion entre los lados derecho e izquierdo implica que, si se encuentran problemas durante la verificacion y validacion, entonces el lado izquiero de la V puede ser ejecutado nuevamente para solucionar el problema. Los planes de prueba son el nexo entre el desarrollo y la verificacion.

.Modelo de prototipos: un prototipo es un producto parcialmente desarrollado que permite a los clientes y desarrollados examinar algunos aspectos del sistema propuesto, y tomar decisiones en base a este. Es una alternativa para tratar de mejor forma la incertibumbre, ambigüedad y volubilidad de los proyectos reales.

Tipos de prototipos:

.Evolutivos: el objetivo es obtener el sistema a entregar. Este permite que todo el sistema o alguna de sus partes se construyan rapidamente para comprender o aclarar aspectos y asegurar que el desarrollador, el usuario y el cliente tengan una compresion unifica tanto de lo que se necesita como de lo que se propone como solucion.

.Descartables: Este no posee funcionalidad, en este tipo de prototipo se utilizan herramientas de modelado. Se basa en construir las partes problematicas y desecharlo. Entiendo que ambos modelos se podrian complementar, porque usar solo el descartable lo veo raro.

.Proyectos candidatos: Estos son desarrollados para poder interactuar con el cliente, estos proyectos tienen un mayor enfasis en la E/S mas que en los detalles algoritmicos. Para asegurar el éxito estos deben: ser un sistema con el que se pueda expermientar, ser comparativamente barato (<10%), debe hacer enfasis en la interzar de usuario y con un equipo reducido.

.Modelo de desarrollo por fases: se desarrolla el sistema de tal manera que puede ser entregado en piezas. Esto implica que existen dos sistemas funcionando en paralelo: el sistema operacional y el sistema en desarrollo.

Tipos:

.Incremental: el sistema es particionado en subsistemas de acuerdo con su funcionalidad. Cada entrega agrega un subsistema.

.Iterativo: este entrega un sistema completo desde el principio y luego aumenta la funcionalidad de cada subsistema con las nuevas versiones.

.Modelo en espiral (Boehm): Este modelo combina las actividades de desarrollo con la gestion del riesgo, trata de mejorar los ciclos de vida clasicos y prototipos, ademas incorpora objetivos de calidad y elimina errores y alternativas no atractivas al comienzo. Permite iteraciones, vuelta atrás y finalizaciones rapidas.

Cada cilo empieza identificando: los objetivos de la porcion correspondiente y las aternativas.

Como restricciones tiene que cada ciclo se completa con una revision que incluye todo el ciclo anterior y el plan para el siguiente.

Cada ciclo en la espiral se divide en cuatro sectores:

.Establecimiento de objetivos: Se identifican restricciones, se traza un plan de gestion, se identifican riesgos.

.Valoracion y reduccion del riesgo: Se analiza cada riesgo identificado y se determinan acciones.

.desarrollo y validacion: se determina el modelo de desarrollo.

.Planeacion: el proyecto se revisa y se toma desiciones para la siguiente fase.

Teoria 8 – Metodologias Agiles:

.Definicion: Una metodologia agil es aquella en la que “se da prioridad a las tareas que dan resultados directos y que reducen la burocracia tanto como sea posible”, adaptandose ademas rapidamente al cambio de proyectos. Tiene como objetivo producir sw de alta calidad con un costo efectivo y en el tiempo apropiado.

Valores: Una buena manera de interpretarlos es asumir que definen preferencias, no alternativas.

.Respuesta al cambio: fomenta la adaptabilidad en lugar de la estricta y rigurosa planificacion.

.Colaboracion con el cliente: se centra en asociarse con los clientes en lugar de negociar contratos.

.Individuos e interacciones: enfatiza la importancia de la comunicación personal y el trabajo en equipo mas que procesos y herramientas.

.SW operante: prioriza la entrega de sw funcional sobre la documentacion extensa.

Principios:

1. La mayor prioridad es satisfacer al cliente a través de fáciles y continuas entregas de software valuable.

2. Los cambios de requerimientos son bienvenidos, aún tardíos, en el desarrollo. Los procesos Ágiles capturan los cambios para que el cliente obtenga ventajas competitivas.

3. Entregas frecuentes de software, desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre una entrega y la siguiente.

4. Usuarios y desarrolladores deben trabajar juntos durante todo el proyecto.

5. Construir proyectos alrededor de motivaciones individuales.

6. Darles el ambiente y el soporte que ellos necesitan y confiar el trabajo dado. El diálogo cara a cara es el método más eficiente y efectivo de intercambiar información entre el equipo de desarrolladores.

7. El software que funciona es la medida clave de progreso.

8. Los procesos ágiles promueven un desarrollo sostenible. Los stakeholders, desarrolladores y usuarios deberían ser capaces de mantener un paso constante indefinidamente.

9. Atención continua a la excelencia técnica y buen diseño incrementa la agilidad.

10. Simplicidad (el arte de maximizar la cantidad de trabajo no dado) es esencial.

11. Las mejores arquitecturas, requerimientos y diseños surgen de la propia organización de los equipos.

12. A intervalos regulares, el equipo reflexiona sobre cómo volverse más efectivo, entonces afina y ajusta su comportamiento en consecuencia.

En la practica estos principios son dificiles de cumplir dado que aunque es atractiva la idea de involucrar al cliente en el proceso de desarrollo, coordinar puede ser complejo; priorizar los cambios tambien podria ser dificil, sobre todo en sistemas donde existen muchos participantes; mantener la simplicidad requiere trabajo adicional, ya que bajo la presion de fechas de entrega es posible que los miembros del equipo carezcan de tiempo para realizar las simplificaciones deseables al sistema; ademas muchas organizaciones, especialmente las grandes compañias, pasan años cambiando su cultura, de tal modo que los procesos se definan y contunen. Para ellas, resulta dificil moverse hacia un modelo de trabajo donde los procesos sean informales y esten definidos por equipos de desarrollo. Y por ultimo elo no contar con mucha documentacion complejizaria a futuro el mantenimiento y evolucion de los sistemas de sw existentes.

Principales Metodologias Agiles:

eXtreme Programming (XP): Es una disciplina de desarrollo de sw basado en los valores de la sencillez, la comunicación, la retroalimentacion, la valentia y el respeto. Su accion consiste en llevar a todo el equipo reunido en la presencia de practicas simples, con suficiente informacion para ver donde estan y ajustar las practicas a su situacion particular. Tiene un desarrollo iterativo e incremental, contando con prubeas unitarias continuas, frecuentemente repetidas y automatizadas. Gran interaccion con el cliente y simplicidad en su codigo.

Las caracterisitcas esenciales son: Historias de usuario, roles, proceso y practicas.

Roles:

.Programador: responsable de decisiones tecnicas, responsalbe de construir el sistema y en XP los programadores diseñan, programan y realizan las pruebas (se engloba a todo junto).

.Jefe de proyecto: organiza y guia las reuniones y asegura condiciones adecuadas para el proyecto.

.Cliente: este es parte del equipo, determina que construir y cuando, ademas estable las pruebas funcionales.

.Entrenador: responsable del proceso, este tiende a estar en 2do plano a medida que el equipo madura.

.Tester: ayuda al cliente con las pruebas funcionales, se asegura de que las pruebas funcionales se superan.

.Rastreador: Metric Man, observa sin molestar.

Proceso: Este posee un ciclo de vida de consta de 6 partes.

1. Exploracion: los clientes plantean HU que son de interes para la primera entrega del producto, el equipo de desarrollo se familiariza con el entorno del proyecto y se construye un prototipo (dura de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnologia).

2. Planificacion: el cliente establece la prioridad de cada HU, los programadores realizan una estimacion de esfuerzo, se acuerda la primera entrega y se determina un cronograma en conjunto con el cliente (dura unos pocos dias).

3. Iteracion: el plan de entrega se compone por iteraciones de no mas de tres semanas, el cliente es quien decide que HU implementar en cada iteracion y al final de la ultima iteracion el sistema estara listo para entrar en produccion. Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado.

4. Produccion: requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusion de nuevas caracteristicas a la version actual, debido a cambios durante esta fase.

5. Mantenimiento: se lleva a cabo mientras la primera version se encuentra en produccion, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Puede requerir nuevo personal dentro del equipo y cambios en su estructura.

6. Muerte: cuando el cliente no tiene mas HU para ser incluidas al sistema, se genera la documentacion final del sistema y no se realizan mas cambios en la arquitectura, la muerte del proyecto tambien ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

Practicas:

Testing: los programadores continuamente escriben pruebas unitarias, las cuales deben correr sobre problemas para que el desarrollo continue.

Refactoring: actividad constante de reestructuracion del codigo con el objetivo de remover duplicacion de codigo, mejorar su legibilidad, simplificarlo y hacerlo mas flexible para facilitar los posteriores cambios.

Programacion de a pares: todo el codigo de produccion es escrito por dos programadores en una maquina.

Propiedad colectiva del codigo: Cualquiera puede cambiar codigo en cualquier parte del sistema en cualquier momento.   
Integracion continua: cada pieza de codigo es integrada en el sistema una vez que este lista. Asi, el sistema puede llegar a ser intregado y construido varias veces en un mismo dia.

.Semana de 40-horas: los proyectos que requieren trabajo extra para intentar cumplir con los plazos suelen al final ser entregados con retraso. En lugar de esto se puede realizar el juego de la planificacion para cambiar el ambito del proyecto o la fecha de entrega.

.Cliente en el lugar de desarrollo: tiene que estar presente y disponible todo el tiempo para el equipo.

.Estandares de codificacion: los programadores escriben todo el codigo de acuerdo con reglas que enfatizan la comunicación a traves del mismo.

SCRUM: es un marco de trabajo utilizado para desarrolloar productos complejos donde se aplican de manera regular un conjunto de buenas practicas para trabajar colaborativamente y obtener el mejor resultado posible de un proyecto. Estas practicas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. “Una manera simple de manejar problemas complejos”. Se realizan entregas parciales y regulares del resultado final del proyecto, priorizadas por el beneficio que aportan al receptor del proyecto.

Principios:

.Eliminar el desperdicio: no generar artefactos, ni perder el tiempo haciendo cosas que no suman valor.

.Construir la calidad con el producto: inyectar la calidad directamente en el codigo desde el inicio.

.Crear conocimiento: en la practica no se puede tener el conocimiento antes de empezar el desarrollo.

.Diferir las decisiones: tomar las decisiones en el momento adecuado, esperar hasta ese momento, ya que uno tiene mas info a medida que va pasando el tiempo. Si se puede esperar, mejor.

.Entregar rapido: debe ser una de las ventajas competitivas mas importantes.

.Respetar a las personas: generar un ambiente donde la gente se sienta respetada y se motive.

.Optimizar el todo: optimizar todo el proceso, ya que el proceso es una unidad, y para lograr el éxito y avanzar, hay que tratarlo como tal.

Roles:

.Propietario: conoce y marca las prioridades del proyecto o producto.

.Jefe: persona que asegura el seguimiento de la metodologia guiando las reuniones y ayudando al equipo ante caulquier problema que pueda aparecer. Su responsabilidad es entre otras, la de hacer de paraguas ante las presiones externas.

.Equipo: son las personas responsables de implementar la funcionalidad o funcionalidades por el Propietario.

.Usuario o cliente: son los beneficiarios del producto y son quienes viendo los progresos, pueden aportar ideas, sugerencias o necesidades.

Artefactos:

Product Backlog: lista maestra que contiene toda la funcionalidad deseada en el producto. Las funcionalidades se encuentran ordenadas por prioridad.

.Sprint Backlog: lista que contiene toda la funcionalidad que el equipo comprometio a desarrollar durante el sprint determinado.

.Burndown Chart: muestra un acumulativo del trabajo hecho, dia-a-dia.

Entre otros.

Proceso: es un solapamiento de fases el cual se puede asemejar a un scrum de rugby, en el cual todos los roles trabajan juntos para lograr un objetivo. Este es iterativo e incremental. Se busca poder atacar todos los problemas que surgen durante el desarrollo del proyecto. Su nombre se debe a que durante los sprints, lo que serian las fases de desarrollo, se solapan, de manera que no es un proceso de cascada por cada iteracion, si no que tenemos todas estas etapas juntas que se ejecutan una y otra vez, hasta que se crea suficiente.

Scrum esta pensado para ser aplicado en proyecto en donde el “caos” es una constante, aquellos proyectos en los que tenemos requerimientos dinamicos, y que tenemos que implementar tecnologia de punta. Esos proyectos dificiles, que con los enfoques tradicionales se hace imposible llegar a buen puerto.

KANBAN:

Es un enfoque Lean de desarrollo de sw agil.

.Enfoque Lean: filosofia que hace hincapie en la eliminacion de residuos o de no valor añadido a traves de la mejora continua para agilizar operaciones. Esta centrado en el cliente, Lean se centra en ofrecer una mayor calidad, reducir el tiempo de ciclo y reducir los costos.

Se presenta como una herramienta de gestion que prescribe solo 3 practicas: limitar el trabajo en curso, visualizar el flujo de trabajo y medir el tiempo promedio de entrega. Esto la convierte en una herramiento simple, pero a la vez muy potente.

.Kanban trata de administrar el flujo de trabajo. Inicialmente, no reemplaza nada en la que la organización haga, simplemente impulsa el cambio. Esta no define ningun rol y no prescribe una secuencia de pasos definidos para llevar a cabo el desarrollo del proyecto. Su implementacion es sencilla (las 3 practicas). Lo habitual es combinar en otras practicas agiles, usando los tableros kanban que son la herramienta agil por excelencia de esta metodologia, donde se visualizan las tareas a realizar, las realizadas y las pendientes.

.Tablero Kanban: a primera vista se asemeja a un proceso en cascada pero en la practica evita explicitamente los problemas de un proceso de estas caracteristicas mediante la aplicación de lotes pequeños de trabajo y el desarrollo incremental. Se compone de columnas que representan los diversos estados que atraviesa un requerimiento durante el proceso de desarrollo. Las tarjetas se mueven de un estado a otro mostrando la evolucion hasta que haya sido aceptado por el cliente.

Cada columna tendra un limite para el trabajo en curso. Cuando una tarea es completada, esta se mueve a la siguiente. De esta forma se crea un espacio libre en la columna actual representando capacidad de trabajo disponible. El equipo entonces toma una tarea terminada desde el estado anterior y la desplaza hacia el estado actual, ya que hay capacidad para procesarla. (Simple de entender).

Teoria 9 – DSBM y Calidad:

MDD:

Desarrollo de sw basado en modelos (MBD) y Desarrollo de sw dirigido por modelos (MDD).

.El adjetivo dirigido en MDD, a diferencia de basado, enfatiza que este paradigma asigna a los modelos un rol central y activo: son al menos tan importantes como el codigo fuente.

MDD promueve enfatizar en puntos clave como:

.Mayor nivel de abstraccion en la especificacion tanto del problema a resolver como de la solucion correspondiente.

.Uso de estandares industriales como medio para facilitar las comunicaciones, la interaccion entre diferentes aplicaciones y productos, y la especializacion tecnologica.

Entre otros.

En el MDD, los modelos pasan de ser entidades contemplativas (interpretadas por los programadores) para convertirse en entidades productivas a partir de las cuales se deriva la implementacion en forma automatica.

.Platform Independent Model (PIM): “Un modelo de un sistema que no contiene informacion acerca de la plataforma o la tecnologia que es usada para implementarlo”.

.Platform Specific Model (PSM): “Especifica el proceso de conversion de un modelo en otro modelo del mismo sistema”.

Transformacion de modelos: “Especifica el proceso de conversion de un modelo en otro modelo del mismo sistema”.

Cada transformacion incluye (al menos): un PIM, un Modelo de la Plataforma, una transformacion, y un PSM.

Los tres pasos principales en el proceso de desarrollo MDD:

Diseño del PIM, Transformacion PIM/PSM y Transformacion PSM/Codigo.

.En general se puede decri que una transformacion consiste en una colección de reglas, las cuales son especificaciones no ambiguas de las formas en que un modelo (o parte de el) puede ser usado para crear otro modelo (o parte de el). El patron MDD es normalmente utilizado sucesivas veces para producir una sucesion de transformaciones.

Pensadolo en java: El PIM seria la estructura de las clases (herencia, atributos, relaciones) y el PSM seria a ese PIM agregarle las funcionalidades (metodos) y luego eso se pasaria a codigo.

.MDD es la evolucion natral de la ingenieria de sw basada en modelos enriquecida mediante el agregado de transformaciones automaticas entre modelos. Si bien MDD define un nuevo paradigma para el desarrollo de sw, sus principios fundamentales no constituyen realmente nuevas ideas sino que son reformulaciones y asociaciones de ideas anteriores.

Beneficios de MDD: incremento en la productividad (modelos y transformaciones), adaptacion a los cambios tecnologicos y de requisitos, consistencia (automatizacion), re-uso (de modelos y transformaciones) , mejoras en la comunicación con los usuarios, modelos que resisten cambios, entre otras.

Calidad:

.Se puede definir a la calidad como: “El grado en el que un conjunto de caracteristicas inherentes cumple con los requisitos”.

.Sistemas de informacion: es el conjunto de personas, datos, procesos y tecnologia de info que interactuan para recopilar, procesar, guardar y proporcionar como salida la info necesaria para brindar soporte a una organización. La importancia de estos en la actualidad genera que las empresas de tecnologia hagan mucho hincapie en los estandares de calidad.

Componentes de la calidad de un sistema de informacion (Stylianou y Kumar):

.calidad de la infraestructura: incluye, por ej, calidad de las redes y sistemas de sw

.calidad de la gestion: incluye el presupuesto, planificacion y programacion

.calidad del servicio: incluye los procesos de atencion al cliente

.calidad de datos: los datos que ingresan en el sistema de informacion

.calidad de la informacion: este item esta relacionado con la calidad de los datos

.calidad del software: incluye aplicaciones de sw construidas, o mantenidas, o con el apoyo de IS.

Y a su vez la calidad de los sistemas de informacion son internos a la calidad de la empresa que tiene como otro componente a la calidad de los procesos de negocios soportados por los sistema de informacion.

.La calidad de sw se ha mejorado significativamente en los ultimos años, en particular por una mayor conciencia de la importancia de la gestion de la calidad y la adopcion de tecnicas de gestion de la calidad para desarrollo en la industria del sw. Esta se divide en calidad del producto obtenido y calidad del proceso de desarrollo.

.La estandarizacion del producto define las propiedades que debe satisfacer el producto sw resultante y la estandarizacion del proceso define la manera de desarrollar el producto sw. “Sin un buen proceso de desarrollo es casi imposible obtener un buen producto”.

¿

Clasificacion de normas y modelos de calidad:

.Calidad de producto de sw, calidad de uso – calidad de datos: ISO/IEC 25000

.Calidad de servicios: ISO/IEC 20000 – ITIL.

.Calidad del proceso de desarrollo: PMBOOK – SWEBOOK – SIX SIGMA, ISO/IEC 12207 – ISO/IEC 15504 (ISO 33000) – ISO/IEC 90003, CMMI.

.Calidad total: TQM – ISO/IEC 9001

.Calidad seguridad de la informacion: ISO/IEC 27001

Entiendo son familias de normas pero nose si es necesario saber cada una y para que son, es raro esta parte.

?

CMM (1993) – CMMI (2000):

.Es un modelo de evaluacion de los procesos de una organización. Es un marco de referencia para desarrollar procesos efectivos y proporciona un marco estructurado para evaluar los procesos actuales de la organización, establecer prioridades de mejora, e implmentar esas mejoras.

CMMI:

.Posee dos vistas que permiten un enfoque diferente según las necesidades de quien vaya a implementarlo.

.Escalonado: centra su foco en la madurez de la organización. Igual que CMM.

.Continuo: enfoca las actividades de mejora y evaluacion en la capacidad de los diferentes procesos. Presenta seis niveles de capacidad. Los niveles de capacidad indican que tan bien se desempeña la organización en un area de proceso individual.

Nivel de Capacidad y nivel de madurez, representacion continua (grafico de barras) y representacion escalonado o por etapas (piramide).

Niveles de madurez:

1. Ambiente NO confiable: proceso impredecible, poco controlado y reactivo.

2. Controles informales: proceso caracterizado por proyectos y frecuentemente reactivo.

3. Controles estandarizados: proceso caracterizado por la organización y proactivo.

4. Controles monitoreados: el proceso es controlado cuantitativamente.

5. Controles optimizados: enfoque en la mejora del proceso.

.La familia ISO 9000 es un conjunto de normas de “gestion de la calidad” aplicables a cualquier tipo de organización con el objetivo de obtener mejoras en la organización y, eventualmente arribar a una certificacion, punto importante a la hora de competir en los mercados globales.

Beneficios de trabajar con un sistema de gestion de calidad (SGC) ISO 9001:

.Rendimiento financiero: aumenta los resultados financieros y la rentabilidad.

.Cumplimiento legal: asegura el cumplimiento de los requisitos legales y del cliente.

.Rendimiento organizacional: mejora la eficiencia y productividad general.

.Toma de decisiones: facilita elecciones informadas y efectivas.

.Satisfaccion del cliente: mejora la satisfaccion y lealtad del cliente.

.SGC – Mejora Continua: Implica un compromiso constante por identificar oportunidades de mejora, implementar cambios y evaluar su impacto. En el contexto de la ingenieria de sw, la mejora continua busca optimizar los procesos de desarrollo, aumentar la calidad del sw y satisfacer mejora las necesidades de los clientes.

Ver cuadro de resumen de la ultima pagina. ES IMPORTANTE, habian varias preguntas en las autoevaluacion sobre esos temas asique tener claro el tema de las familias de normas y para que sirve y a que estan dirigidas cada una.