Capítulo 1

Diana Ortega Barron/181080479

Este primer capítulo se ocupa de la obtención, análisis, especificación y validación, así como la gestión de los requisitos durante todo el ciclo de vida del software. Los requisitos tienen otras cualidades además de las características del comportamiento de sí mismo que expresan en ciertas técnicas, los ejemplos comunes incluyen una tasa de prioridad para permitir compensaciones ante recursos infinito y un valor del progreso y que el proyecto sea supervisado. Pueden ser investigados más a fondo según si son requisitos de funcionamiento, requisitos de funcionamiento, de seguridad, confiabilidad o algún otro requisitos de muchos tipos de software, ya que estos discuten en su totalidad la calidad del software.

En cuanto al proceso de los requisitos, este procesos hace referencia a los requisitos del software orientado a las cinco subzonas restantes y demostrando cómo el proceso de los requisitos encaja con el proceso de ing. en software.

Mateos Ramón Rebeca
No.Control: 181080442

En la mayoría de las profesiones de la ingeniería, la especificación se refiere a la asignación de valores o limites a los objetivos de diseño de un producto. En ingeniería de software normalmente los requisitos se refieren a la producción de un documento que pueda ser revisado sistemáticamente y para los sistemas complejos son aquellos que involucran sustanciales no software componentes, se pueden producir tres diferentes: definición del sistema, requisitos del sistema y requisitos del software.

Este documento se conoce como el usuario documento de requisitos o concepto de operaciones documento, define los requisitos de sistema de alto nivel de la perspectiva del dominio, sus lectores incluyen representantes de los usuarios del sistema.

Este documento enumera los requisitos del sistema junto con antecedentes de información sobre los objetivos generales del sistema, este puede incluir modelos conceptuales diseñado para ilustrar el contexto, el uso, escenarios y principales entidades de dominio, como los flujos de trabajo.

Un desarrollador de sistema con software sustancias y componentes que no son de software seria el avión, porque separa la descripción de los requisitos del

sistema y el sistema se especifica los requisitos de software porque se derivan de los sistemas y después los requisitos para los componentes de software se especifican.

Establece que la base del acuerdo entre clientes y contratistas sobre lo que el software el producto debe hacer tan bien como lo que no se espera hacer. Una evaluación rigorosa de los requisitos de diseño puede comenzar y reducir el diseño posterior y eso debe dar una base realista para estimar costos, riesgos y cronogramas del producto.

Pueden ser validado para asegurar que el software ah entendido los requisitos, es importante verificar que un documento de requisitos cumple con los estándares de la empresa y que es comprensible y concreto. Las notaciones formales ofrecen la importante ventaja de permitir probar las dos ultimas propiedades, diferentes partes interesadas incluidos los representantes del cliente y el desarrollador debe revisar los documentos. Cuando sea practico los requisitos individuales son también sujeto a la gestión de la configuración.

Es el medio de validación más común mediante inspección o revisión de los requisitos, se le asigna un grupo de revisores para buscar errores o suposiciones erróneas, falta de claridad y desviación de la practica estándar.

La creación de los prototipos es un medio para validar la interpretación de software asi como para obtener nuevos requisitos, al igual que con la provocación, hay un rango de técnicas de creación y una serie de puntos en el proceso donde la validación del prototipo puede ser apropiado y una de las ventajas es que facilitan la interpretación del software, suposiciones del ingeniero y cuando sea necesario dar u8na retroalimentación sobre porque están equivocados. La volatilidad de un requisito se define después de la creación de prototipos entre la parte interesada y el ingeniero de software por lo tanto para seguridad critica y crucial la creación de prototipos realmente ayudaría.

Es necesario validar la calidad de los modelos desarrollados durante el análisis, los modelos de objetos es útil realizar un análisis estético para verificar que las rutas de comunicación existen entre objetos que en las partes interesadas.

Una propiedad esencial de software debería ser posible para validar que el producto terminado lo satisfaga. Los requisitos que no se pueden validar son los deseos, por lo tanto, una tarea importante es planificar como verificar cada requisito. En la mayoría de los casos, diseñar pruebas de aceptación lo hacen para determinar como los usuarios finales suelen realizar negocios utilizando el sistema. Para ser validados primero deben

de ser analizados y descompuesto hasta el punto en el que puedan ser expresados cuantitativamente.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

Las Consideraciones prácticas son aquellas que contienen el primer nivel de descomposición, puede describirse en una secuencia de actividades, la cual contiene un proceso de requisitos el cual abarca todo un ciclo de vida de dicho software. Tiene una gestión del cambio y mantenimiento de los requisitos en un estado que refleja con precisión el software que se va a construir, o que se ha construido, son clave para el éxito del proceso de ingeniería de software.

Podemos decir que no en todas las organizaciones cuentan con una cultura de documentación y gestión de requisitos, es muy común ver empresas que son emergentes dinámicas, pero estas son impulsadas por una fuerte "visión del producto" y recursos limitados, para ver su documentación y su gestión de requisitos la ven como innecesaria, conforme la empresa se va expandiendo, va aumentando la base de clientes, y su producto va evolucionando, descubren que necesitan recuperar los requisitos que motivaron las características del producto para evaluar el impacto de los cambios propuestos. Por tanto, los requisitos la documentación y la gestión de cambios son fundamentales para el éxito de cualquier proceso de requisitos.

En la Industria del Software se tiene una presión generalizada, ya que los ciclos de desarrollo cada vez son más cortos, y esto es particularmente pronunciado en altamente competitivo, sectores impulsados por el mercado. Además, la mayoría de los proyectos están limitados de alguna manera por su entorno, y muchas son actualizaciones o revisiones de software existente donde la arquitectura es un hecho. Para implementar el proceso de requisitos como lineal, proceso determinista en el que los requisitos de software se obtienen de las partes interesadas, se establecen como referencia, se asignan y se entregan al software.

Lo que viene siendo gestión de cambios es una parte de suma importancia para la gestión de requisitos. El tema se describe como la función de la gestión de cambios, los procedimientos que deben implementarse y el análisis que se debe aplicar a los cambios propuestos.

Los requisitos deben consistir no sólo en una especificación de lo que se requiere, sino también en información auxiliar, que ayuda a

gestionar e interpretar los requisitos. Los atributos de los requisitos deben definirse, registrarse y actualizarse a medida que evoluciona el software en desarrollo o mantenimiento.

Las herramientas para lidiar con los requisitos de software se dividen en dos categorías: herramientas para modelar y herramientas para gestionar requisitos. Las herramientas de gestión de requisitos suelen admitir una variedad de actividades, incluida la documentación, el seguimiento y la gestión de cambios, y han tenido un impacto significativo en la práctica.

ALUMNO: ROSAS TRINIDAD SERGIO ANTONIO

No.CONTROL: 181080399

En este capítulo introduce al lector un ambiente general del paradigma experimental aplicado a la Ingeniería de Software (IS), así como presentan algunas investigaciones hechas en México que han utilizado este enfoque de investigación. También se comentan los enfoques ingenieriles en el desarrollo y mantenimiento de productos software datan desde la década de los años 60s, ya que hasta en la década de los 80s que la comunidad académica comenzó a adoptar y emplear enfoques de investigación para estudiar de manera más rigurosa los diferentes aspectos y problemáticas involucradas en el desarrollo de software.

Desde sus inicios en la década de 1940, escribir software ha evolucionado hasta convertirse en una profesión que se ocupa de cómo crear software y mejorar su calidad. La calidad puede referirse a cuán mantenible es el software, su estabilidad, velocidad, usabilidad, con probabilidad, legibilidad, tamaño, costo, seguridad y número de fallas o "bugs", así como, entre muchos otros atributos, a cualidades menos medibles como elegancia, concisión y satisfacción del cliente.

Puedo decir que la investigación en ingeniería de software tiene como fin generar conocimiento nuevo en el ámbito de esta disciplina, así como modificarlo de tal manera que este pueda ser incorporado en la práctica diaria del desarrollo de software. Esto con el fin de mejorar los plazos de entrega, la ejecución del presupuesto asignado, así como mejorar la calidad del producto software a desarrollar o mantener.

Capítulo 2

Diana Ortega Barron/181080479

En cuanto al diseño de software, una vez que se han establecido los requisitos del software, el diseño es la primera de tres actividades técnicas: diseño, codificación y prueba. Cada actividad transforma la información de forma que al final se obtiene un software validado, el diseño es técnicamente la parte central de la ingeniería del software y durante el diseño se desarrollan, revisan y se documentan los refinamientos progresivos de las estructuras de datos, de la estructura del programa y de los detalles procedimentales. El diseño da como resultado representaciones cuya calidad puede ser evaluada. Mediante algunas metodologías de diseño se realiza el diseño de datos, el diseño arquitectónico y el diseño procedimental. El diseño de datos transforma el modelo de campo de información, creado durante el análisis, en las estructuras de datos que se le implementaran al software. El diseño arquitectónico define las relaciones entre los principales elementos estructurales del programa. El diseño procedimental transforma los elementos estructurales en una descripción procedimental del software.

Mateos Ramón Rebeca
No.Control: 181080442

En esta sección se incluye una serie de análisis de calidad y temas de evaluación que están específicamente relacionados con el diseño de software.

Varias herramientas y técnicas pueden ayudar a analizar y evaluar la calidad del software, las revisiones pueden ser informales y formales para determinar la calidad de artefactos de diseño, estas pueden evaluar la seguridad, la cual puede ayudar para su instalación y uso.

Las medidas se pueden utilizar para evaluar cuantitativamente o estimar varios aspectos de un software como por ejemplo el tamaño, estructura o calidad, la mayoría de las medidas que se han propuesto dependen del enfoque que se utiliza para producir el diseño. Estas se clasifican en funciones que se obtienen analizando funcional descomposición representado por un diagrama de estructura. Medidas orientadas a objetos que se representan mediante un diagrama que puede tomar varias medidas.

Hay diferentes tipos de notaciones para el diseño de un software de artefactos y algunos de ellos se utilizan para describir la estructura de diseño y otras para el diseño de software, ambas se utilizan principalmente durante el diseño detallado y otras se pueden utilizar para ambos propósitos. El diseño se logra utilizando varias anotaciones que se clasifican para describir la estructura de la estética.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

Como podemos darnos cuenta hay que saber que estrategias y métodos hay en el diseño de software, para esto existen varias estrategias generales las cuales son útiles para el proceso de diseño, incluyen las estrategias de dividir y conquistar y de refinamiento paso a paso, los métodos son más específicos en el sentido de que generalmente proporcionan un conjunto de notaciones que se utilizarán con el método, una descripción del proceso que se utilizará al seguir el método y un conjunto de pautas para utilizar el método. Estos métodos son útiles como marco común para los equipos de ingenieros de software.

El diseño orientado contiene ciertas funciones, es uno de los métodos más clásicos de diseño de software, donde existe una descomposición la cual se centra en identificar las principales funciones del software y luego elaborarlas y refinarse, el diseño estructurado se utiliza generalmente después del análisis estructurado, produciendo así (entre otras cosas) diagramas de flujo de datos y descripciones de procesos asociados. Lo que es el diseño orientado a objetos se ha propuesto numerosos métodos de diseño de software basados en objetos, ha ido evolucionando desde el diseño orientado a objetos a mediados de la década 1980, donde la herencia y el polimorfismo juegan un papel clave, en el campo del diseño basado en componentes. Aunque las raíces del diseño OO se derivan del concepto de abstracción de datos, el diseño está impulsado por la responsabilidad que se ha propuesto como un enfoque alternativo al diseño OO.

La estructura de datos tiene un diseño centrado comienza a partir de las estructuras de datos que se llega a manipular en un programa en lugar de las funciones que se realizan. El ingeniero de software primero describe las estructuras de datos de entrada y salida y luego desarrolla la estructura de control del programa basándose en estos diagramas de estructura de datos. También existen otros métodos y para los diseños de dichos métodos se necesitan herramientas para el software, estas herramientas se pueden utilizar para respaldar la creación de artefactos de diseño de software.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

En el área del conocimiento de los requisitos de software (KA), se refiere al análisis, especificación y validación de los requisitos de software.

Donde nos están demostrando ampliamente que el simple hecho de no hacer las cosas bien este proceso puede traer consecuencias malas en el desarrollo de cualquier producto de software.

Uno de los requisitos del software es una característica que se debe exhibir para solucionar un cierto problema en el mundo real. Se convierte en una combinación compleja de requisitos entregados por parte de los usuarios implicados dentro del desarrollo de la solución, teniendo en cuenta que pueden corresponder a diferentes niveles jerárquicos, ambientes e intereses. Es importante también que cada requisito los podamos comprobar, pensando también en las implicaciones que esto puede llevar.

También nos comentan de todo lo que necesita el software para funcionar desde el punto de vista de hardware, software, recurso humano, información, instalaciones, servicios, etc.

Se dice que se tiene que tener ciertas habilidades con las que debe contar el ingeniero de software, apoyado en herramientas que le ayuden a contextualizar el software.

También la Negociación de los requisitos que llega a esta etapa cuando hay incompatibilidad en dos o más requisitos y los solicitantes son distintos dice que el ingeniero de software debe reunirlos y tomar la decisión más conveniente para el correcto funcionamiento de la solución.

Existen medios que utiliza el ingeniero de software para manifestar lo que entendió y para facilitar la corrección, eliminación o adición de requisitos. Como se evidencia, la ventaja de hacerlo es grande pero el costo puede ser alto.

Hay Pruebas de Aceptación Son aquellas que se le aplican a cada requisito para determinar si el producto final lo satisface o no.

Debe haber total planeación para aplicar estas pruebas y hacerlo de manera cuantitativa. Como conclusión para este capítulo puede afirmarse que los requisitos de software deben tomarse como una práctica seria, asignándole el tiempo y los recursos necesarios que permitan continuar con un proceso ordenado para llegar a obtener un software de calidad.

Capítulo 3

Diana Ortega Barron/181080479

El diseño se define en [IEEE610.12-90) como "el proceso para definir la arquitectura, los componentes, los interfaces, y otras características de un sistema o un componente y el resultado de este proceso. Visto como proceso, el diseño del software es la actividad del ciclo de vida de la cual los requisitos del software se analizan para producir una descripción de la estructura interna del software que servirá como la base para su construcción. Más exacto, un diseño del software (el resultado) debe describir la arquitectura del software, en cómo la descomposición del software, la organización de los componentes, y los interfaces entre los mismos componentes. Debe también describir los componentes en un nivel de detalle que permita su construcción. El diseño del software desempeña un papel importante en el desarrollo de software: permite que la Ingeniería del software produzca los diversos modelos para la solución que se pondrá en desarrollo. Podemos analizar y evaluar estos modelos para determinar si o no permitirán que se satisfaga los requisitos.

Mateos Ramón Rebeca

No.Control: 181080442

Un entorno de desarrollo proporciona una instalación para programadores de software, integrando un conjunto de desarrollo de herramientas, las opciones pueden afectar la eficiencia y calidad del software. Los IDE modernos ofrecen características como la compilación y detección de errores desde el editor, integración de control de código y compilación. Un constructor de GUI incluye un editor visual para que el desarrollador diseñe formularios y ventanas, algunos pueden generar la fuente código correspondiente al diseño de la GUI visual hay que las actuales siguen el estilo impulsado por eventos. Algunos GUI modernos, proporcionan una GUI integrada y otros son independientes. Las pruebas unitarias suelen estar automatizadas, los desarrolladores pueden usar herramientas de prueba unitaria y marcos para ampliar y crear entorno de pruebas.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Nos especifican el término construcción de software que se refiere a la creación detallada de software funcional mediante una combinación

de codificación, verificación, pruebas unitarias, pruebas de integración y depuración. El área de conocimiento de Construcción de software (KA) está vinculada a todos los demás KA, pero está más fuertemente vinculada al Diseño de software y las Pruebas de software porque el proceso de construcción de software implica un diseño y pruebas de software importantes.

Comenta sobre el proceso que utiliza la salida del diseño y proporciona una entrada para las pruebas. Dice que los límites entre el diseño, la construcción y las pruebas variarán según los procesos del ciclo de vida del software que se utilicen en un proyecto, aunque se puede realizar algún diseño detallado antes de la construcción, gran parte del trabajo de diseño se realiza durante la actividad de construcción. Por lo tanto, el KA de construcción de software está estrechamente relacionado con el KA de diseño de software.

Durante la construcción, los ingenieros de software, tanto la prueba unitaria como la integración, prueban su trabajo. Por lo tanto, el KA de construcción de software también está estrechamente relacionado con el KA de pruebas de software.

Es importante saber que la construcción de software generalmente produce la mayor cantidad de elementos de configuración que deben administrarse en un proyecto de software por lo tanto, el KA de construcción de software también está estrechamente vinculado al KA de gestión de configuración de software. Si bien la calidad del software es importante en todos los KA, el código es el resultado final de un proyecto de software y, por lo tanto, el KA de calidad del software está estrechamente relacionado con el KA de construcción de software.

También está relacionado con la gestión de proyectos, en la medida en que la gestión de la construcción puede presentar desafíos considerables.

Capítulo 4

Diana Ortega Barron/18108O479

El término construcción del software hace referencia a la creación detallada de software operativo y significativo, por medio de una combinación de codificación, verificación, pruebas unitarias, pruebas de integración y depuración. El Área de Conocimiento de la Construcción del Software está vinculada a todas las otras KAs (Areas de Conocimiento), más fuertemente al Diseño del Software y a las pruebas del Software. Esto se debe a que el proceso mismo de construcción del software cubre tanto el diseño significativo de software como las actividades de pruebas. También utiliza las salidas del diseño y proporciona una de las entradas para las pruebas, consistiendo estas actividades en el diseño y en las pruebas, y en este caso no en las KAs. Las fronteras detalladas entre el diseño, la construcción y las pruebas (si es que existen) varian dependiendo de los procesos de ciclo de vida del software utilizados en un proyecto. A pesar de que se pueda realizar parte del diseño detallado antes de la construcción, mucho del trabajo del diseño se lleva a cabo durante la actividad misma de la construcción. Por lo que el KA de Construcción del Software está vinculado muy de cerca al KA de Diseño del Software. Por medio de la construcción los ingenieros del software realizan tanto pruebas unitarias, como pruebas de integración de su trabajo. De tal manera que el KA de Construcción del Software está también vinculada de cerca al KA de Pruebas del Software. La construcción del software, por lo general, produce el mayor número de elementos de configuración que se necesitan gestionar en un proyecto de software (archivos de código fuente, contenido, casos de pruebas, etc.). De este modo, el KA de Construcción del

Mateos Ramón Rebeca

No.Control: 18108O442

Las pruebas requieren de tareas laboriosas, de ejecutar numerosas ejecuciones de programas y manejo una gran

cantidad de información, las herramientas apropiadas pueden aliviar la carga de las operaciones administrativas y tediosas y hacerlos menos propensos a errores. La selección de herramientas depende de evidencia diversa, como opciones de desarrollo, objetivos de evaluación, facilidades de ejecución. Es posible que no exista una herramienta que satisfaga necesidades particulares, por lo que un conjunto de herramientas podría ser una opción.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Se señala que una actividad que permite evaluar y mejorar la calidad del producto con el fin de detectar fallas y corregir errores. Las pruebas del software consisten en verificar el comportamiento de un programa dinámicamente a través de un grupo finito de casos de prueba, debidamente seleccionados.

Se ha ido cambiando la percepción de que las pruebas de software se realizan únicamente al final del proceso de creación de código fuente, siendo muy útil hacerlo en todas las etapas del desarrollo del software; esto permite corregir errores y detectar fallas de fondo, a tiempo.

En el capítulo se dice que en los últimos años, la visión de las pruebas de software se ha convertido en una visión constructiva. Las pruebas ya no se consideran una actividad que comienza solo después de que se completa la fase de codificación con el propósito limitado de detectar fallas.

Las pruebas de software son, o deberían ser, generalizadas durante todo el ciclo de vida de desarrollo y mantenimiento. De hecho, la planificación de las pruebas de software debe comenzar con las primeras etapas del proceso de requisitos de software, y los planes y procedimientos de prueba deben desarrollarse sistemática y continuamente, y posiblemente refinarse, a medida que avanza el desarrollo de software y estas actividades de planificación y diseño de pruebas proporcionan información útil para los diseñadores de software y ayudan a resaltar posibles debilidades, como descuidos / contradicciones de diseño u omisiones / ambigüedades en la documentación.

Capítulo 5

Diana Ortega Barron/181080479

La apreciación de las pruebas del software ha evolucionado hacia una forma más constructiva. Ya no se asume que realizar pruebas es una tarea que empieza solamente cuando la fase de programación se ha completado, y que tiene el único propósito de detectar errores. Las pruebas del software se ven ahora como una actividad que debería estar presente durante todo el proceso de desarrollo y mantenimiento y es en sí misma una parte importante de la construcción del producto. Es más, la planificación de las pruebas debería empezar en las primeras etapas del proceso de requisitos, mientras que los planes y procedimientos de pruebas deberían desarrollarse y posiblemente refinarse sistemáticamente según avanza el desarrollo. La planificación de las pruebas y las propias actividades de diseño constituyen una información muy útil que ayuda a los diseñadores de software a identificar debilidades potenciales tales como elementos del diseño que han pasado desapercibidos, contradicciones de diseño, u omisiones o ambigüedades en la documentación).

Mateos Ramón Rebeca

No.Control: 181080442

Nos habla sobre las herramientas que son importantes en el mantenimiento de software, donde se este modificando, algunos de los ejemplos incluyen las herramientas de ingeniería inversa que ayudan al proceso que es trabajado hacia atrás desde un producto que ya existe para crear artefactos, especificaciones y diseño.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

El proceso de desarrollo de software debe satisfacer los requerimientos planteados, una vez en operación el proceso de cubrimiento de defectos, operación y cambio de ambiente debe darse en esta etapa, la fase de mantenimiento empieza con un periodo de garantía y de soporte post- implementación, pero el mantenimiento del software ocurre mucho antes. Aunque la etapa de mantenimiento del software no ha tenido el grado de atención que se debe este tipo de desarrollo de software ya está empezando a cambiar ya que muchos

errores graves han ocurrido por no prestarle la atención que se merece.

Al igual los esfuerzos de desarrollo de software dan como resultado la entrega de un producto de software que satisface los requisitos del usuario.

Como consecuencia, el producto de software debe cambiar o evolucionar. Una vez en funcionamiento, se descubren defectos, los entornos operativos cambian y surgen nuevos requisitos de usuario. La fase de mantenimiento del ciclo de vida comienza después de un período de garantía o de la entrega de soporte posterior a la implementación, pero las actividades de mantenimiento ocurren mucho antes.

Capítulo 6

Diana Ortega Barron/181080479

El mantenimiento de software es una parte del ciclo de vida de software. Sin embargo, históricamente, no ha recibido el mismo grado de atención que otras fases del ciclo de vida Históricamente, el desarrollo de software ha tenido un perfil mucho más alto que el mantenimiento de software en la mayor parte de organizaciones. Esto ha cambiado ahora, las organizaciones se esfuerzan en exprimir al máximo su inversión de desarrollo de software por lo que desean que el software funcione tanto tiempo como sea posible. Las preocupaciones sobre el efecto 2000, enfocó una atención significativa en la fase de mantenimiento de software, y el paradigma Open Source ha atraído la atención a la cuestión de mantener software desarrollado por otros.

Mateos Ramón Rebeca

No.Control: 181080442

Una auditoria de software es un examen independiente de un producto de trabajo o un conjunto de productos de trabajo para evaluar el cumplimiento de especificaciones, estándares, acuerdos contractuales u otros criterios. Un propósito del software es garantizar que el elemento de software auditado es consistente con sus especificaciones que rigen y la salida del software son las actividades de verificación y validación en la

calidad del software.

El contexto se refiere a la distribución de un elemento y su construcción es la actividad de combinar versiones correctas de la configuración en un programa ejecutable para entrega a un cliente u otro destinatario como la actividad de prueba por sistemas con hardware. El software se crea utilizando versiones particulares de herramientas de apoyo como lo son los compiladores y podría ser necesario reconstruir una copia exacta de un elemento de configuración. Las versiones de software abarcan la identificación, empaque y entrega de elementos de un producto ejecutable, documentaciones notas de versión y datos de configuración.

Se necesita una capacidad de herramienta para soportar esas funciones de gestión de versiones, es útil tener una conexión con la capacidad de la herramienta, apoyar el proceso de solicitud de cambio para mapear el contenido de la liberación a los SCR que se han recibido.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

Como sabemos existen varias herramientas y esta Herramienta es de gestión de configuración de software, es muy útil clasificar, esta herramienta tiene tres divisiones en cuyos términos del alcance, para así brindar un buen soporte: está el soporte individual, soporte relacionado con proyectos y soporte de procesos en toda la empresa.

La primera herramienta es la de soporte individual la cual es muy apropiada y generalmente son suficientes para realizar organizaciones pequeñas o grupos de desarrollo y no tienen variantes de los productos de software. Esto incluye:

• Herramientas de control de versiones: sirven para rastrear, documentar y almacenar elementos de configuración individuales.

• Herramientas de manejo de compilación: en su forma más simple, dichas herramientas compilan y vinculan una versión ejecutable del software.

• Herramientas de control de cambios: soportan principalmente el control de solicitudes de cambio y notificación de eventos (por ejemplo, cambios de estado de solicitudes de cambio, hitos alcanzados).

Las herramientas de soporte relacionadas con proyectos apoyan principalmente la gestión del espacio de trabajo para equipos de desarrollo e integradores; por lo general, pueden admitir entornos de desarrollo distribuidos. Estas herramientas son apropiadas para organizaciones medianas y grandes con variantes de sus productos de software y desarrollo paralelo, pero sin requisitos de certificación.

Las herramientas de soporte de procesos de toda la empresa pueden, por lo general, automatizar partes de un proceso de toda la empresa, proporcionando soporte para la gestión del flujo de trabajo, los roles y las responsabilidades. Pueden manejar muchos elementos, datos y ciclos de vida. Estas herramientas se suman al apoyo relacionado con el proyecto al respaldar un proceso de desarrollo más formal, incluidos los requisitos de certificación.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Este capítulo menciona que un sistema puede definirse como la combinación de elementos que interactúan organizados para lograr uno o más propósitos establecidos como la configuración de un sistema son las características funcionales y físicas del hardware o software o es lo que según se establece en la documentación técnica o se logra en un producto, también consideran como una colección de versiones específicas de hardware o elementos de software con procedimientos de construcción específicos para cumplir un propósito en particular.

La gestión de la configuración (CM), entonces, es la disciplina de identificar la configuración de un sistema en distintos puntos en el tiempo con el fin de controlar sistemáticamente los cambios en la configuración y mantener la integridad y trazabilidad de la configuración a lo largo del ciclo de vida del sistema y una disciplina que aplica la dirección y vigilancia técnica y administrativa para: identificar y documentar las características funcionales y físicas de un elemento de configuración, controlar los cambios en esas características, registrar e informar el estado de implementación y procesamiento de cambios y verificar el cumplimiento de los requisitos especificados.

La gestión de la configuración del software es un proceso del ciclo de vida del software de apoyo que beneficia la gestión de proyectos, las actividades de desarrollo y mantenimiento, las actividades de garantía de calidad, así como a los clientes y usuarios del producto final. Los conceptos de gestión de la configuración se aplican a todos los elementos que se van a controlar, aunque existen algunas diferencias en la implementación entre la que brindan garantía de que los productos y procesos de software en el ciclo de vida del proyecto cumplen con sus requisitos especificados mediante la planificación, la implementación y la realización de un conjunto de actividades para proporcionar la confianza adecuada de que la calidad es incorporado en el software.

Las actividades de SCM son la administración y planificación del proceso de SCM, la identificación de la configuración del software, el control de la configuración del software, la contabilidad del estado de la configuración del software, la auditoría de la configuración del software y la administración y entrega de versiones del software.

Capítulo 7

Diana Ortega Barron/18108O479

Para planificar un proceso de la SCM para un proyecto se necesita comprender el contexto de organización y la relación entre los distintos elementos de la organización. La SCM interacciona con otras actividades o elementos de la organización. Los elementos de la organización responsables de los procesos de soporte de la ingeniería del software se pueden estructurar de diferentes formas. Aunque la responsabilidad de realizar algunas de las tareas de la SCM se podría asignar a otra de las partes de la organización como por ejemplo la organización de desarrollo, normalmente es un elemento definido de la organización o un individuo especialmente designado quien tiene la responsabilidad general de la SCM. El software se desarrolla frecuentemente como una parte de un sistema mayor que contiene elementos de hardware y firmware. En ese caso, las actividades de la SCM

Mateos Ramón Rebeca

No.Control: 18108O442

Una fase importante de un proyecto o un ciclo de desarrollo iterativo llega a su fin cuando todos los planes y procesos hayan sido promulgado y completado. Se ha completado un proyecto, una fase o una iteración y logro satisfactorio de la finalización,

los requisitos del software se pueden confirmar como satisfechos o no y el grado de consecución de los objetivos puede ser determinado.

Una medida eficaz se ha convertido en uno de los pilares de madurez organizacional, la medición puede ser aplicado a organizaciones, proyectos, procesos y productos del trabajo.

Evaluar los productos de información y la medición proceso contra evaluación especificada criterios y determinar las fortalezas y debilidades de los productos de información o proceso, respectivamente.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

Al parecer las herramientas de gestión de la ingeniería de software se llegan a utilizar muy a menudo para proporcionar la visibilidad y control de los procesos de dicha gestión de software. Existe una tendencia la cual tiene conjuntos integrados de herramientas de la ingeniería de software, se utilizan para un largo proyecto para planificar, recopilar, registrar, monitorear, controlar y reportar información sobre proyectos y productos. Puede haber diversas herramientas las cuales están en las siguientes categorías:

Herramientas de planificación y seguimiento de proyectos. Las herramientas de planificación y seguimiento de proyectos se pueden utilizar para estimar el esfuerzo y el costo del proyecto y para preparar los cronogramas del proyecto.

Las herramientas de seguimiento se pueden utilizar para realizar un seguimiento de los hitos del proyecto.

Herramientas de gestión de riesgos. Las herramientas de gestión de riesgos se pueden utilizar para realizar un seguimiento de la identificación, estimación y seguimiento de riesgos.

Herramientas de comunicación. Las herramientas de comunicación pueden ayudar a proporcionar información oportuna y consistente a las partes interesadas relevantes involucradas en un proyecto. Estas herramientas pueden incluir cosas como notificaciones por correo electrónico y transmisiones a miembros del equipo y partes interesadas.

Herramientas de medición. Las herramientas de medición utilizadas para recopilar, analizar e informar los datos de medición del proyecto pueden basarse en hojas de cálculo desarrolladas por miembros del equipo del proyecto o empleados de la organización.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Este capítulo puede ser definido como las actividades de gestión de la aplicación, planeación, coordinación, medición, monitorización, control y reportes para asegurar el desarrollo y mantenimiento del software como sistemático, disciplinado y cuantificable ya que esto es un aspecto muy importante en la administración y medición de la ingeniería del software.

Aquí se destacan tres niveles: Gestión y organización de la infraestructura, gestión de proyectos, y control y planeación del programa de medidas. Los aspectos de la gestión de la organización son importantes en términos del impacto en la ingeniería del software y en las políticas de gestión, esas políticas pueden ser influenciadas por los requerimientos de un software efectivo, mantenimiento y desarrollo. Un número de políticas específicas deben ser establecidas para una efectiva gestión en la ingeniería del software y el nivel organizacional.

Se comenta acerca de evaluar las mediciones de este proceso ya que se debe llevar a cabo con un criterio de evaluación específico para determinar las fuerzas y debilidades de los productos, se puede hacer por medio de un proceso de auditoría interna o externa y debe incluir una retroalimentación a los usuarios. Se deben identificar las mejoras potenciales y costos y beneficios de estas, comunicarlas a la persona encargada para su revisión y aprobación.

Capítulo 8

Diana Ortega Barron/181080479

Los ingenieros del software deben entender los aspectos de gestión que se encuentran influidos de modo singular por el software, y además conocer sus aspectos más generales, incluso en los primeros cuatro años tras graduarse, como está marcado en la Guía. La cultura y comportamiento organizacional y la gestión comercial funcional en términos de consecución de metas, aportan la gestión en cadena, la publicidad, la venta y la

distribución, todas ellas influyen, aunque sea indirectamente, en el proceso de ingeniería del software de una organización.
El enfoque de este conjunto de actividades se centra en la determinación eficaz de los requisitos del software por medio de varios métodos de inducción y la valoración de la viabilidad del proyecto desde distintos puntos de vista. Una vez que se ha establecido la viabilidad, la tarea pendiente dentro de este proceso es la especificación de la validación de requisitos y del cambio de procedimientos.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

En el capítulo se examinan dos niveles desde lo técnico y desde el meta-nivel o nivel de implementación, valoración, medición y gestión también existen varios significados sobre el proceso de la ingeniería del software puede verse como una sola manera de realizar el proceso o como muchas maneras que es lo que se quiere y se debe llegar a hacer ya que el proceso de la ingeniería abarca muchos factores, finalmente un tercer significado se refiere al conjunto actual de actividades realizadas dentro de una organización que se puede ver como un solo proceso. Los procesos de ingeniería del software son considerados de gran importancia, el objetivo es gestionar nuevos y mejores procesos.

También comentan acerca de la medición de los procesos y los productos, aunque puede resultar compleja la medición a la ingeniería del software existen varios aspectos que son fundamentales para la medición y análisis, estas mediciones se pueden realizar para apoyar los procesos de implementación y cambio o evaluar consecuencias de estos.

La medición de un producto software incluye principalmente, la medición del tamaño del producto, la estructura del producto y la calidad del producto.

También es importante tener en cuenta la medición del tamaño, de la estructura y de la calidad. Para garantizar la calidad en los resultados de la medición es necesaria la medición efectiva de los programas para proveer resultados exitosos.

Capítulo 9

Diana Ortega Barron/181080479

Aquí se describe la situación en la que los procesos se despliegan por primera vez (por ejemplo, introduciendo un proceso de inspección en un proyecto o un método que cubra todo el ciclo de vida), y donde se cambian los procesos actuales (por ejemplo, introduciendo una herramienta u optimizando un procedimiento). A esto también se le puede denominar proceso de evolución. En ambos casos hay que modificar las prácticas actuales. Si resulta que se extienden las modificaciones, puede que también sean necesarios cambios en la cultura organizacional.

Este tópico incluye el conocimiento relacionado con la infraestructura del proceso de ingeniería del software. Para establecer procesos de ciclo de vida del software, es necesario que la adecuada infraestructura esté en su lugar, es decir que los recursos estén al alcance de la mano (personal competente, herramientas y financiación) y que se hayan asignado responsabilidades. El que se hayan completado estas tareas, indica el compromiso con la gestión y propiedad del esfuerzo del proceso de ingeniería del software. Puede que haya que establecer diversos comités, tales como un comité de dirección que supervise el esfuerzo del proceso de ingeniería del software.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO
No. CONTROL: 181080399

En este capítulo el tema principal son los instrumentos asistidos por ordenador que se requieren para ayudar a los procesos de ciclo de vida del software y nos ayuda a reducir la carga cognoscitiva enfocándonos más en los aspectos creativos del proceso.

También los métodos imponen la estructura a la actividad de la ingeniería del software con el objetivo de hacerla sistemática, por lo general proporcionan notación y vocabulario para comprobar tanto el proceso como el producto, aunque existen numerosos materiales sobre los instrumentos de apoyo en la ingeniería del software, los textos sobre las técnicas sobre estos instrumentos son relativamente escasos, una dificultad es el alto costo que

representa un cambio de instrumento de software en general, los instrumentos y métodos cubren ciclos de vida completos por esto es tan complicado hacer un cambio.

También se cubren métodos que implican el prototipado de software y es subdivida en estilos de prototipado, objetivos y técnicas de evaluación. Se deben incluir aspectos como estilos de prototipado, objetivos del prototipado, y las técnicas para la evaluación del prototipo propuesto.

Capítul 10

Diana Ortega Barron/181080479

Los cinco primeros asuntos del subárea de los Instrumentos de Ingeniería de Software corresponden a las cinco primeras áreas del conocimiento de la Guía (Exigencias de Software, el Diseño de Software, la Construcción de Software, Pruebas de Software, y el Mantenimiento de Software). Los cuatro siguientes asuntos corresponden a las áreas de conocimiento restantes (la Dirección de Configuración de Software, la Dirección de la Ingeniería de Software, el Proceso de Ingeniería de Software, y la calidad de Software). Proporcionan un asunto adicional, dirigiendo áreas como las técnicas de integración de instrumento que son potencialmente aplicables a todas las clases de instrumentos

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

En este capítulo se abordan los aspectos relativos a la calidad del software los cuales trascienden en cualquier ciclo de vida, en esta guía se describen un conjunto de métodos para alcanzar la calidad, en este caso se tratarán las técnicas estáticas es decir aquellas que no requieren la ejecución del software para su evaluación mientras las dinámicas cubren los aspectos de calidad en las pruebas del software como los Fundamentos de Calidad del Software que aquí se definen formalmente los aspectos a tratar y la manera como un ingeniero de software debería entender y adoptar los conceptos y características de

calidad y su relevancia en el desarrollo o mantenimiento de software.

Se comentan algunos aspectos de calidad que deben estar inherentes desde el momento mismo de los requerimientos, así como la medición y criterios de aceptación que evalúan estas características.

Modelos y Características de Calidad usada en las características que cada una posee niveles jerárquicos diferentes, hay tres modelos relacionados con la calidad en un producto de software. La mejora de calidad da a la tarea de ser mejorada cada vez más gracias a un proceso iterativo de mejora continua que requiere control de dirección, control y retroalimentación de muchos procesos

Capítulo 11

Diana Ortega Barron/181080479

El concepto de calidad no es tan simple como parece, para un ingeniero de productos hay muchas cualidades deseadas relevantes para una perspectiva determinada de un producto, para que esto pueda ser tratado y determinado en el tiempo las exigencias de producto son puestas por escrito. Las características de calidad pueden requerirse o no, o se pueden requerir en, el coste de calidad puede segmentarse en el coste de prevención, el coste de apreciación, el coste de fracaso interno, y el coste de fracaso externo.

La motivación latente tras un proyecto de software es el deseo de crear un software que tiene valor, y este valor puede o no puede ser cuantificado como un coste. El cliente tendrá en mente algún coste máximo, a cambio del cual espera que se cumpla el objetivo básico del software. El cliente también puede tener alguna expectativa en cuanto a la calidad del software. En ocasiones los clientes pueden no haber estudiado detenidamente las cuestiones de calidad o sus gastos relacionados. ¿La calidad es meramente decorativa, o es consustancial al software? Si la respuesta se sitúa en un punto intermedio, como es casi siempre el caso, es cuestión de hacer del cliente parte del proceso de decisión y concienciar totalmente tanto de los costes como de los beneficios. Idealmente, la mayor parte de estas decisiones serán adoptadas en el proceso de requerimientos de software.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO
No. CONTROL: 181080399

Este capítulo se centra en relacionar cada una de las disciplinas con la Ingeniería del Software.

Es específico en lo que respecta a determinar las temáticas de cada una de ellas en este capítulo se centra en relacionar cada una de las disciplinas con la Ingeniería del Software.

Es específico en lo que respecta a determinar las temáticas de cada una de ellas ya que el área de conocimiento de la práctica profesional de ingeniería de software se ocupa del conocimiento, las habilidades y las actitudes que los ingenieros de software deben poseer para practicar la ingeniería de software de manera profesional, responsable y ética. Debido a las aplicaciones generalizadas de los productos de software en la vida social y personal, la calidad de los productos de software puede tener un impacto profundo en nuestro bienestar

personal y armonía social. Los ingenieros de software deben manejar problemas de ingeniería únicos, produciendo software con características y confiabilidad conocidas. Este requisito requiere ingenieros de software que posean un conjunto adecuado de conocimientos, habilidades, capacitación y experiencia en la práctica profesional.

Capítulo 12

Diana Ortega Barron/181080479

Para circunscribir la Ingeniería del Software, es necesario identificar las disciplinas con las que la Ingeniería del Software comparte un límite común. Este capítulo identifica, en orden alfabético, esas Disciplinas Relacionadas. Por supuesto, las Disciplinas Relacionadas también comparten varios límites en común entre ellas mismas.

Usado como fuente reconocida y consensuada, este capítulo identifica para cada Disciplina Relacionada:

Una definición informativa (cuando sea factible) y un listado de áreas de conocimiento.

Mateos Ramón Rebeca

No.Control: 181080442

Los proyectos y productos de ingeniería de software no son precisos sobre los objetivos que deberían lograr, se indican los requisitos, pero con un valor marginal de agregar un poco mas de funcionalidad y no se puede medir, al principio se relaciona el valor marginal con el costo marginal y proporciona orientación para determinar los criterios cuando un entregable es suficiente bueno para ser entregado.

La fricción económica es todo lo que mantiene a los mercados a tener competencia perfecta, involucra distancia, costo de entrega, regulaciones restrictivas o información imperfecta. Esto es difícil para nuevos competidores para iniciar negocios y competir.

Un ecosistema típico tiene productos y consumidores que agregan los recursos consumidos, el usuario final es una organización que utiliza el producto para mejorarlo. Un ecosistema de software es un proveedor de una aplicación que

trabaja con empresas que realizan la instalación y el soporte en diferentes regiones, ninguno podría existir sin el otro.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Yo pienso que respecto a la economía de la ingeniería de software trata de tomar decisiones relacionadas con la ingeniería de software en un contexto empresarial.

Se dice que el éxito de un producto, servicio y solución de software depende de una buena gestión empresarial sin embargo, en muchas empresas y organizaciones, las relaciones comerciales de software con el desarrollo y la ingeniería de software siguen siendo bajas. Esta área de conocimiento proporciona una descripción general de la economía de la ingeniería de software.

Se comenta que la economía es el estudio del valor, los costos, los recursos y su relación en un contexto o situación determinados. En la disciplina de la ingeniería de software, las actividades tienen costos, pero el software resultante en sí mismo tiene atributos económicos también y la economía de la ingeniería de software proporciona una forma de estudiar los atributos del software y los procesos de software de una manera sistemática que los relaciona con medidas económicas. Estas medidas económicas pueden pensarse y analizarse al tomar decisiones que están dentro del alcance de una organización de software y aquellas dentro del alcance integrado de todo un negocio de producción o adquisición.

Capítulo 13

Diana Ortega Barron/181080479

El alcance del área de conocimiento de Fundamentos de la Computación (KA) abarca el entorno operativo y de desarrollo en el que el software evoluciona y se ejecuta. Debido a que ningún software puede existir en el vacío o ejecutarse sin una computadora, el núcleo de dicho entorno es la computadora y sus diversos componentes. El conocimiento sobre la computadora y sus principios subyacentes de hardware y software sirve como marco sobre el que se ancla la ingeniería de software. Por lo tanto, todos los ingenieros de software deben tener un buen conocimiento de Computing Foundations KA.
La solución de problemas se refiere al pensamiento y las actividades realizadas para responder o derivar una solución a un problema. Hay muchas formas de abordar un problema, y cada una de ellas emplea diferentes herramientas y utiliza diferentes procesos. Estas diferentes formas de enfocar los problemas se amplían y definen gradualmente y finalmente dan lugar a diferentes disciplinas. Por ejemplo, la ingeniería de programas informáticos se centra en la solución de problemas utilizando computadoras y programas informáticos.
Si bien los diferentes problemas justifican diferentes soluciones y pueden requerir diferentes instrumentos y procesos, la metodología y las técnicas utilizadas en la solución de problemas siguen algunas directrices y a menudo pueden generalizarse como técnicas de solución de problemas.

Mateos Ramón Rebeca

No.Control: 181080442

Una vez que el programa esta codificado y compilado el siguiente paso es la depuración que es un proceso metódico de encontrar y reducir la cantidad de errores o fallas en un programa, su propósito es encontrar porque un programa no funciona o produce un resultado o salida incorrecta.

La depuración implica muchas actividades y puede ser estático, dinámico o post mortem, toma la forma de revisión de código, mientras que la depuración dinámica toma forma de revisión de código mientras que la depuración dinámica toma la forma de seguimiento y esta estrechamente asociado con las pruebas.

Puede ser compleja y tediosa al igual que la programación pero también es muy creativa, por lo que es necesario alguna ayuda de herramientas por la depuración dinámica, y se utilizan para

habilitar al programador para que monitoree la ejecución de un programa.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

Los algoritmos guían las operaciones de las computadoras y consisten en una secuencia de acciones compuestas para resolver un problema. Un algoritmo es cualquier procedimiento computacional bien definido que toma algún valor o conjunto de valores como entrada y que produce algún valor o conjunto de valores como salida. Podemos ver que los atributos de los algoritmos son muchos e incluyen modularidad, corrección, fácil mantenimiento, funcionalidad, solidez, es fácil de entender por las personas, tiene un tiempo de programación, es simple y extenso. El atributo comúnmente enfatizado cuenta con un desempeño o eficiencia, hay un cierto punto, donde la eficiencia determina si un algoritmo es factible o impráctico. Por ejemplo, un algoritmo que tarda cien años en finalizar es prácticamente inútil e incluso se considera incorrecto.

El análisis de algoritmos es el estudio teórico del rendimiento de los programas informáticos y el uso de recursos; hasta cierto punto, determina la bondad de un algoritmo. Este análisis abstrae los detalles particulares de una computadora específica, existe tres tipos básicos de análisis, en el análisis del peor de los casos, se determina el tiempo máximo o los recursos requeridos por el algoritmo en cualquier entrada de tamaño, en el segundo análisis de caso promedio, se determina el tiempo esperado o los recursos requeridos por el algoritmo sobre todas las entradas de tamaño y por último tenemos el tercer tipo de análisis es el análisis del mejor de los casos, en el que se determina el tiempo o los recursos mínimos requeridos por el algoritmo en cualquier entrada de tamaño n. Entre los tres tipos de análisis, el análisis de casos promedio es el más relevante pero también el más difícil de realizar.

El diseño de algoritmos generalmente sigue una de las siguientes estrategias: fuerza bruta, divide y vencerás, programación dinámica y selección codiciosa. La estrategia de fuerza bruta es en realidad una no estrategia. Intenta exhaustivamente todas las formas posibles de abordar un problema. Si un problema tiene solución, esta estrategia está garantizada para encontrarla; sin embargo, el gasto de tiempo puede ser demasiado elevado. La estrategia de divide y vencerás

mejora la estrategia de fuerza bruta al dividir un gran problema en problemas más pequeños y homogéneos.

La estrategia de programación dinámica mejora la estrategia de divide y vencerás al reconocer que algunos de los sub-problemas producidos por la división pueden ser los mismos y así evita resolver los mismos problemas una y otra vez.

La estrategia de selección codiciosa mejora aún más la programación dinámica al reconocer que no todos los subproblemas contribuyen a la solución del gran problema.

Ian Sommerville escribió, "un sistema es una colección intencionada de componentes interrelacionados que trabajan juntos para lograr algún objetivo". Un sistema puede ser muy simple e incluir solo unos pocos componentes, como un bolígrafo de tinta, o más bien complejo, como un avión. Dependiendo de si los seres humanos son parte del sistema, los sistemas se pueden dividir en sistemas técnicos basados en computadora y sistemas socio técnicos.

El sistema es más que la simple suma de sus partes. Por tanto, las propiedades de un sistema no son simplemente la suma de las propiedades de sus componentes. En cambio, un sistema a menudo exhibe propiedades que son propiedades del sistema como un todo. Estas propiedades se denominan propiedades emergentes porque se desarrollan solo después de la integración de las partes constituyentes en el sistema. Las propiedades emergentes del sistema pueden ser funcionales o no funcionales. Las propiedades funcionales describen las cosas que hace un sistema y las propiedades no funcionales describen cómo se comporta el sistema en su entorno operativo.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Es importante saber el alcance del área de conocimiento de Fundamentos de la Computación ya que abarca el entorno operativo y de desarrollo en el que el software evoluciona y se ejecuta, debido a que ningún software puede existir en el vacío o ejecutarse sin una computadora, el núcleo de dicho entorno es la computadora y sus diversos componentes.

El conocimiento sobre la computadora y sus principios subyacentes de hardware y software sirve como marco sobre el que se ancla la ingeniería de software.

La mayoría de los temas discutidos en el capítulo también son temas de discusión en cursos básicos impartidos en programas de pregrado y posgrado en ciencias de la computación. Dichos cursos incluyen programación, estructura de datos, algoritmos, organización de computadoras, sistemas operativos, compiladores, bases de datos, redes, sistemas distribuidos, etc. Por lo tanto, al desglosar temas, puede ser tentador descomponer los Fundamentos de Computación de acuerdo con estas divisiones que se encuentran a menudo en los cursos relevantes.

CAPÍTULO 14

Diana Ortega Barron/1810080479

Un conjunto es una colección de objetos que se llaman elementos del set, ya que un conjunto se puede representar enumerando sus elementos entre llaves. Una relación es una asociación entre dos conjuntos de información. Por ejemplo, consideremos un conjunto de los residentes de una ciudad y sus números de teléfono. El emparejamiento de nombres con el teléfono correspondiente. los números son una relación. Este emparejamiento se solicita para toda la relación. En el ejemplo que se está considerando, para cada par, el nombre viene primero seguido del número de teléfono o al revés.

Una función es una relación de buen comportamiento. Una relación R (X, Y) se comporta bien si la función mapea cada elemento del dominio establece X en un solo elemento del rango establecido y consideremos el conjunto de dominios X como un conjunto de personas y deje que el conjunto de rango Y almacene sus números de teléfono. Esto significa que, si bien todas las funciones relaciones, no todas las relaciones son funciones. En caso de una función dada una x, se obtiene una y exactamente una y para cada par ordenado (x,y).

En la escuela vemos varios tipos de lógica pero en esta ocasión nada más veremos un tipo de lógica, pero primero vamos a definir que la lógica básica.

La lógica básica es una disciplina la cual aborda aspectos más sencillos, la cual hablaré es sobre la lógica propositiva o proporcional, esta se define como una proposición que es una declaración que es verdadera o falsa, pero no ambas. Consideremos las frases declarativas para las que es significativo asignar cualquiera de los dos valores de estado: verdadero o falso.

Predicar la lógica es una serie de plantillas de frases verbales que describen una propiedad de los objetos o una relación entre los objetos representados por las variables. Por ejemplo, en la frase, La flor

es roja, la plantilla es roja es un predicado. Describe la propiedad de una flor. El mismo predicado puede ser usado en otras oraciones también.

Alumna: Mateos Ramón Rebeca

No.Control: 181080442

La probabilidad es la descripción matemática de aleatoriedad. Definición básica de probabilidad y la aleatoriedad se ha definido en la sección 4 de este KA. Aquí, comencemos con los conceptos detrás distribución de probabilidad y probabilidad discreta.
Un modelo de probabilidad es una descripción matemática de un fenómeno aleatorio que consta de dos partes: un espacio muestral S y una forma de asignar probabilidades a eventos. El espacio muestral define el conjunto de todos los resultados posibles, mientras que un evento es un subconjunto de un espacio muestral que representa un posible resultado o un conjunto de resultados.
Una variable aleatoria es una función o regla que asigna un número a cada resultado. Básicamente, es solo un símbolo que representa el resultado de una experimentar.
Por ejemplo, sea X el número de caras cuando el experimento lanza una moneda n veces de manera similar, sea S la velocidad de un automóvil registrada en un detector de radar.
Los valores de una variable aleatoria pueden ser discretos o continuos según el experimento.
Una variable aleatoria discreta puede contener todo lo posible resultados sin perder ninguno, aunque puede llevar una cantidad infinita de tiempo.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

En este capítulo 14 pude notar que existe un sistema informático el cual es abstraído como si fuera un mapa que va de un estado a otro y está impulsado por entradas, el sistema puede considerarse como dicha función de transición, la S es el conjunto de estados de I, O las cuales tienen su propio significado, el conjunto de estados S es finito (no infinito), el sistema se denomina máquina de estados finitos (FSM).

Una máquina de estados finitos (FSM) es una abstracción matemática compuesta de un número finito de estados y de transiciones entre esos estados. Si el dominio S × I es razonablemente pequeño, entonces se puede especificar T explícitamente utilizando diagramas similares a un diagrama de flujo para ilustrar la forma en que la lógica flujos para diferentes entradas. Sin embargo, esto es práctico sólo para las máquinas que tienen una muy pequeña capacidad de información.

Un FSM tiene una memoria interna finita, una entrada que lee los símbolos en una secuencia y una a la vez, y una función de salida.

La función que tiene un FSM comienza desde un estado de inicio, pasa a través de transiciones dependiendo de la entrada a diferentes estados, y puede terminar en cualquier estado válido. Sin embargo, sólo unos pocos de todos los estados marcan un flujo de operación exitoso. Estos se llaman estados de aceptación.

Como ya había dicho la I e O tienen su propia definición, pero también existe una máquina de estados finitos la cual se define formalmente como M = (S, I, O, f, g, s0).

S es el conjunto de estados; I es el conjunto de símbolos de entrada; O es el conjunto de símbolos de salida; f es la función de transición de estados; g es la función de salida; y s0 es el estado inicial.

Existe una gramática de un lenguaje natural la cual nos dice que hay una combinación de palabras que hace una oración válida. A diferencia de los lenguajes naturales, un lenguaje formal se especifica por un conjunto bien definido de reglas para las sintaxis. Las frases válidas de un lenguaje formal pueden ser descritas por una gramática con la ayuda de estas reglas, llamadas reglas de producción.

El lenguaje formal significa a un conjunto de palabras o cuerdas de longitud finita sobre algún alfabeto finito, y una gramática especifica las reglas para la formación de estas palabras o cuerdas. El conjunto de palabras válidas para una gramática constituye el lenguaje para la gramática. Así pues, la gramática G es cualquier definición matemática compacta y precisa de un idioma L, en contraposición a una mera enumeración de todas las frases legales del idioma o ejemplos de esas frases.

Hay varios tipos de gramáticas las cuales son las siguientes:

- V es el vocabulario, es decir, el conjunto de palabras.

- $T \subseteq V$ es un conjunto de palabras llamadas terminales.

- $S \in N$ es una palabra especial llamada símbolo de inicio.

- P es el conjunto de reglas de producción para sustituir un fragmento de frase por otro.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

Se habla de los profesionales del software y como conviven con los programas. En un lenguaje muy simple, uno puede programar solo para algo que sigue una lógica bien entendida .

El área de conocimiento de Fundamentos matemáticos ayuda a los ingenieros de software a comprender esta lógica, que a su vez se traduce al código del lenguaje de programación. Las matemáticas que son el enfoque principal en este son bastante diferentes de la aritmética típica, donde los números se tratan y se discuten.

La lógica y el razonamiento son la esencia de las matemáticas que un ingeniero de software debe abordar.

Se comenta que las matemáticas, en cierto sentido, son el estudio de sistemas formales. La palabra "formal" está asociada a la precisión, por lo que no puede haber una interpretación ambigua o errónea del hecho. Por lo tanto, las matemáticas son el estudio de todas y cada una de las verdades ciertas sobre cualquier concepto. Este concepto puede ser tanto de números como de símbolos, imágenes, sonidos, videos, casi cualquier cosa.

En resumen, no solo los números y las ecuaciones numéricas están sujetos a precisión si no por el contrario, un ingeniero de software necesita tener una abstracción precisa en un dominio de aplicación diverso.

El capítulo cubre técnicas básicas para identificar un conjunto de reglas para el razonamiento en el contexto del sistema en estudio. Todo lo que se pueda deducir siguiendo estas reglas es una certeza absoluta dentro del contexto de ese sistema. En este se definen y discuten técnicas que pueden representar y hacer avanzar el razonamiento y juicio de un ingeniero de software de una manera precisa y por lo tanto matemática.

El lenguaje y los métodos de lógica que se discuten aquí nos permiten describir pruebas matemáticas para inferir de manera concluyente la verdad absoluta de ciertos conceptos más allá de los números.

CAPÍTULO 15

ALUMNA: Rebeca Mateos Ramón
No.Control: 181080442

El modelado es parte del proceso de abstracción utilizado para representar algunos aspectos de un sistema. Simulación utiliza un modelo del sistema y proporciona unos medios de realizar experimentos diseñados con ese modelo para comprender mejor el sistema, es comportamiento y relaciones entre subsistemas, así como analizar aspectos del diseño. La creación de prototipos es otro proceso de abstracción donde una representación parcial (que captura aspectos de interés) del producto o sistema. Un prototipo puede ser una versión inicial del sistema, pero carece de la funcionalidad completa de la versión final.

Un modelo es siempre una abstracción de algún o artefacto imaginado. Los ingenieros utilizan modelos de muchas formas como parte de su resolución de problemas ocupaciones. Algunos modelos son físicos, como una construcción en miniatura a escala de un puente o edificio. Otros modelos pueden no ser representaciones, como un dibujo CAD de un engranaje o un modelo matemático para un proceso.

Todos los modelos de simulación son una especificación de la realidad. Un tema central en la simulación es abstraer y especificar una simplificación adecuada de la realidad. Desarrollar esta abstracción es de vital importancia, como especificación errónea de la abstracción invalidará los resultados de la simulación ejercicio. Construir un prototipo de un sistema es otro proceso de abstracción. En este caso, una versión inicial del sistema está construida, a menudo mientras el sistema está siendo diseñado. Esto ayuda a los diseñadores determinar la viabilidad de su diseño. Hay muchos usos para un prototipo, incluidos la obtención de requisitos, el diseño y refinamiento de una interfaz de usuario para el sistema, validación de requisitos funcionales, etc.

Los estándares proporcionan requisitos, especificaciones, directrices o características que deben ser observados por ingenieros para que los productos, procesos, y los materiales tengan niveles aceptables de calidad. Las cualidades que brindan varios estándares pueden ser los de seguridad, confiabilidad u otras características del producto. Los estándares se consideran críticos para los ingenieros y se espera que los ingenieros estar familiarizado con las normas adecuadas y utilizarlas

en su disciplina.

En muchos proyectos de ingeniería, conocer y comprender los estándares aplicables es fundamental y la ley puede incluso exigir el uso de normas. En estos casos, los estándares a menudo representan requisitos mínimos que deben cumplir el esfuerzo y por lo tanto son un elemento en las limitaciones impuesto a cualquier esfuerzo de diseño.

Nombre: Diana ortega barron
181080479

IEEE define la ingeniería como la aplicación de un enfoque sistemático, disciplinado y cuantificable a estructuras, máquinas, productos, sistemas o procesos. Este capítulo describe algunas de las habilidades y técnicas fundamentales de la ingeniería que son útiles para un ingeniero de software. El enfoque está en temas que apoyan a otros KA mientras se minimiza la duplicación de temas cubiertos en otras partes de este documento.

A medida que la teoría y la práctica de la ingeniería de software maduran, es cada vez más evidente que la ingeniería de software es una disciplina de ingeniería que se basa en conocimientos y habilidades comunes a todos disciplinas de ingeniería.

ALUMNA: Laura Durán Aguilar

No. CONTROL: 181080392

El análisis de causa raíz (ACR) es un proceso diseñado para investigar e identificar por qué y cómo ha ocurrido un evento que no se ha deseado. Las causas de raíz son causas subyacentes. Debe de haber un investigador el cual tiene que tratar de identificar las causas específicas del evento en el que ha ocurrido. Existe un objetivo principal del análisis de causa raíz que sirve para prevenir la recurrencia del evento indeseable. Mientras tanto, cuanto más específico pueda ser el investigador sobre el motivo por el que ocurrió el evento o problema, será más fácil prevenir la recurrencia.

Se llegan a utilizar muchos enfoques tanto para el control de calidad como para el análisis de las causas profundas. El primer paso en cualquier esfuerzo de análisis de causa raíz es identificar el problema real. Se utilizan técnicas tales como los diagramas de declaración-representación, por qué-por qué, el método de revisión, los diagramas de estado actual y estado deseado, y el enfoque de "ojos frescos" para identificar y perfeccionar el problema real que debe abordarse.

ALUMNO:ROSAS TRINIDAD SERGIO ANTONIO

No. CONTROL: 181080399

IEEE define la ingeniería como "la aplicación de un enfoque sistemático, disciplinado y cuantificable a estructuras, máquinas, productos, sistemas o procesos"

En este capítulo describe algunas de las habilidades y técnicas fundamentales de ingeniería que son útiles para un ingeniero de software. La atención se centra en temas que apoyan a otros mientras se minimiza la duplicación de temas cubiertos en otras partes de este documento.

A medida que madura la teoría y la práctica de la ingeniería de software, es cada vez más evidente que la ingeniería de software es una disciplina de ingeniería que se basa en conocimientos y habilidades comunes a todas las disciplinas de ingeniería.

Esta área de conocimiento de Fundamentos de Ingeniería (KA) se ocupa de los fundamentos de ingeniería que se aplican a la ingeniería de software y otras disciplinas de ingeniería. Los temas de esta incluyen métodos y técnicas experimentales como análisis estadístico, medición, diseño de ingeniería, modelado, creación de prototipos y simulación, estándares, y análisis de la causa raíz. La aplicación de este conocimiento, según corresponda, permitirá a los ingenieros de software desarrollar y mantener el software de manera más eficiente y eficaz.