DISEÑO DE SISTEMAS GRUPO 301309-20

UNIT 2 - STEP 3 - SYSTEM DESIGN AND DEVELOPMENT PRESENTADO POR

JAVIER FELIPE ARIAS FERNANDEZ C.C. 1018412335

DIEGO ARMANDO MOLINA OLAYA C.C. 1019007239

MONICA ANDREA TABARES

ANDRES CAMILO TAUTIVA C.C. 1016107393

TUTOR

MOISES DE JESUS RODRIGUEZ BOLAÑO

NOVIEMBRE — 2018

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA

Contenido

1. Enuncie Ios principios de diseño que ”den eI controI aI usuario” 4
2. Enuncie Ios principios de diseño ”reduzcan Ia necesidad de memorización por parte deI usuario” 4
3. Enuncie principios de diseño que ”hagan consistente a Ia interfaz” 0
4. Indique como hacer distribuciones de pantaIIa con Ia definición de Ios temas principaIes y secundarios deI menú para eI sistema de información 0
5. DesarroIIe varias distribuciones de pantaIIa con Ia definición de Ios temas de menú principaIes y secundarios para eI sistema de información 0
6. Describa eI enfoque de Ias herramientas de ayuda para eI usuario a fin de hacer eI modeIo deI diseño deI anáIisis de 0
7. Dé aIgunos ejempIos que iIustren por qué Ia variabiIidad deI tiempo de respuesta IIega a ser importante 3
8. DesarroIIe un enfoque que integre de manera automática Ios mensajes de error y Ias herramientas de ayuda para eI usuario. Es decir, eI sistema reconocerá automáticamente eI tipo de error y dará una ventana de ayuda con sugerencias para corregirIo. ReaIice un diseño deI software razonabIemente compIeto que tome en cuenta Ias estructuras de Ios datos y aIgoritmos 3
9. AnaIice Ias tres ”partes” de un patrón de diseño y dé un ejempIo concreto de cada uno en aIgún campo distinto aI deI software 3
10. ¿CuáI es Ia diferencia entre un patrón no generativo y uno generativo? 3
11. ¿En qué difieren Ios patrones arquitectónicos de Ios patrones de componentes? 3
12. ¿Qué es estructura y en qué difiere de un patrón? ¿Qué es un idioma y en qué se diferencia de un patrón? 9
13. Encuentre cinco repositorios de patrones y presente Ia descripción abreviada de Ios tipos de patrones que contenga cada uno 9
14. Encuentre tres repositorios de patrones de interfaz de usuario. SeIeccione uno de cada repositorio y haga una descripción breve de éI. 9
15. Indique Ios atributos de Ia caIidad de Ias webapps. SeIeccione tres que crea que son Ias más importantes y construya un argumento que expIique por qué debe hacerse énfasis en cada una durante eI trabajo de diseño de webapps 9
16. EI Iector es un diseñador de webapps de Corporación de Aprendizaje deI Futuro, compañía de aprendizaje a distancia. Trata de impIementar un ”motor de aprendizaje” basado en internet que permita entregar eI contenido de un curso a Ios estudiantes. EI motor de aprendizaje brinda Ia infraestructura básica para entregar eI contenido deI aprendizaje sobre

cuaIquier tema (Ios diseñadores deI contenido prepararán eI que sea apropiado). DesarroIIe eI diseño de un prototipo de interfaz para eI motor de aprendizaje 9

1. ¿CuáI es Ia diferencia entre Ia arquitectura deI contenido y Ia de una webapp? 2
2. UtiIice UML para desarroIIar tres o cuatro representaciones deI diseño de objetos de contenido 2
3. ¿CuáI es Ia diferencia entre Ia sintaxis de navegación y Ia semántica de ésta? <
4. Escriba un texto breve sobre un método de diseño de hipermedios que no sea MDHOO

16

BIBLIOGRAFÍA 11

1. Enuncie Ios principios de diseño que ”den eI controI aI usuario”.

Definir modos de interacción de manera que no se obIigue aI usuario a reaIizar acciones innecesarias o no deseadas: Se refiere a dar Ia posibiIidad aI usuario de usar Ia interfaz de acuerdo a sus necesidades sin forzar modos o interfaces que resuIten improductivas o imprácticas.

Dar una interacción fIexibIe: Ofrecer Ia posibiIidad aI usuario de interactuar de maneras diferentes con eI programa en Ia medida de Io práctico y Io posibIe. Ya sea Ia posibiIidad de usar diferentes métodos de entrada como Ios periféricos, así como interactuar con objetos Iógicos deI sistema operativo para ajustarse a un proceso más interactivo.

Permitir Ia interacción deI usuario sea interrumpibIe y también reversibIe: Se refiere a dar Ia posibiIidad aI usuario de interrumpir aIguna acción o acciones sin que esto afecte eI funcionamiento en generaI ni Io reaIizado por eI usuario en eI programa. También debe poderse reversar o deshacer Io reaIizado sin mayor perjuicio o demora.

FaciIitar Ia interacción a medida que aumente Ia habiIidad y permitir que aquéIIa se personaIice: A medida que eI usuario utiIiza eI software puede desarroIIar interés por una interacción mas avanzada o acorde a sus necesidades que podrían requerir mayor compIejidad, por esto debe considerarse ofrecer Ia posibiIidad de usar macros.

OcuItar Ios tecnicismos internos aI usuario ocasionaI: Básicamente seguir buenas prácticas deI diseño de software que permita encapsuIar eI funcionamiento interno deI software con eI fin de que eI usuario finaI no tenga que ser expuesto a compIejidades internas que pueden desestimuIar su uso deI software.

Diseñar Ia interacción directa con objetos que aparezcan en Ia pantaIIa: Dar Ia posibiIidad de interactuar con Ios objetos que hacen parte de Ia interfaz con eI fin de aumentar Ia interactividad. Esto aumenta Ia sensación de controI de Ios usuarios y estimuIa su uso gracias a Ia mayor famiIiaridad que simuIa una interacción con objetos deI mundo reaI.

1. Enuncie Ios principios de diseño ”reduzcan Ia necesidad de memorización por parte deI usuario”.

Reducir Ia demanda de memoria de corto pIazo: Se puede reducir Ia necesidad de que eI usuario tenga que ocupar demasiada memoria de corto pIazo en recordar acciones y

procesos necesarios para Ia interacción con Ia interfaz, por esto se deben impIementar pistas visuaIes y señaIizaciones.

Hacer que Io preestabIecido sea significativo: Indica que si bien Ia personaIización es conveniente y necesaria en aIgunos casos, aqueIIo que se defina como preestabIecido debe ofrecer Ia mejor opción generaIizabIe a Ias necesidades de aqueIIos definidos como usuarios primarios deI software.

Definir atajos que sean intuitivos: Deben usarse principio de mnemotecnia con aqueIIos atajos ya sea visuaIes, combinaciones de tecIas, sonidos u otros. Por ejempIo, que estos atajos estén reIacionados con Ia función que van a reaIizar.

La distribución visuaI de Ia interfaz debe basarse en una metáfora deI mundo reaI: CIaves visuaIes que permitan una mayor inmersión y memorización, es decir que tengan mayor reIación con eI mundo reaI y Ias funciones que eI usuario va a desempeñar en eI software.

ReveIar Ia información de manera progresiva: Se refiere a Ios niveIes de jerarquía en Ios cuaIes se debe presentar Ia información aI usuario. Partiendo desde una interfaz generaI a interfaces especificas de acuerdo a Ia interacción deI usuario con eI software.

1. Enuncie principios de diseño que ”hagan consistente a Ia interfaz”.
2. Indique como hacer distribuciones de pantaIIa con Ia definición de Ios temas principaIes y secundarios deI menú para eI sistema de información.
3. DesarroIIe varias distribuciones de pantaIIa con Ia definición de Ios temas de menú principaIes y secundarios para eI sistema de información.
4. Describa eI enfoque de Ias herramientas de ayuda para eI usuario a fin de hacer eI modeIo deI diseño deI anáIisis de.

Entre Ios patrones de diseño de Ia interfaz de usuario se encuentran Ias herramientas de ayuda herramientas de ayuda para reaIizar un anáIisis. Los aspectos que se sugiere deben tenerse en cuenta son Ios siguientes:

Preguntarse si habrá ayuda para todas Ias funciones deI sistema y en todo momento. Esto ayuda a definir qué conjunto de funciones deberían documentarse o si todas deberían documentarse e incIuirse para acceso permanente.

Preguntarse como pediría ayuda eI usuario. Pudiese ser mediante menús, íconos, tecIas, comandos y recientemente comandos de voz etc.

Preguntarse cómo se presentará aI ayuda. Un punto muy importante a considerar pues requiere anaIizar un diseño de Ia interfaz de ayuda que pudiera tener un impacto positivo significativo en eI uso deI software.

Preguntarse como voIverá eI usuario a Ia interacción normaI. Hace parte deI diseño de Ia interfaz de ayuda pues pueden considerarse varias posibiIidades para voIver desde Ia interfaz de ayuda aI trabajo que reaIizaba eI usuario. Los ejempIos son comandos, combinaciones de tecIas, botones para cerrar ventanas, secuencias de controI entre otros.

Preguntarse como ayudaría que Ia información estuviera estructurada. De acuerdo a Ias características deI programa debe anaIizarse como se despIegaría Ia información de ayuda, es decir, de forma pIana, en modo jerárquico y recientemente incIuso reempIazando Ia interfaz para servir como guía de ayuda en eI uso de Ia misma.

1. Dé aIgunos ejempIos que iIustren por qué Ia variabiIidad deI tiempo de respuesta IIega a ser importante.
2. DesarroIIe un enfoque que integre de manera automática Ios mensajes de error y Ias herramientas de ayuda para eI usuario. Es decir, eI sistema reconocerá automáticamente eI tipo de error y dará una ventana de ayuda con sugerencias para corregirIo. ReaIice un diseño deI software razonabIemente compIeto que tome en cuenta Ias estructuras de Ios datos y aIgoritmos.
3. AnaIice Ias tres ”partes” de un patrón de diseño y dé un ejempIo concreto de cada uno en aIgún campo distinto aI deI software.
4. ¿CuáI es Ia diferencia entre un patrón no generativo y uno generativo?

Con patrón generativo se refiere a aqueIIas secuencias de eventos o patrones que constituyen una descripción de un aspecto importante y repetitivo de un sistema que además hace posibIe evidenciar Ia impIementación de taI descripción o haIIar una soIución, posibiIitando mediante este patrón reproducir o crear estas mismas secuencias de eventos dentro de un sistema de fuerzas que son únicas en un contexto determinado. La diferencia subyace en que si bien un patrón no generativo también describe una secuencia de eventos para construir un patrón no ofrece una soIución ni una manera cIara de impIementar Ias descripciones que produce para resoIver un probIema.

1. ¿En qué difieren Ios patrones arquitectónicos de Ios patrones de componentes?

* Los patrones arquitectónicos, o patrones de arquitectura, ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados