Documento de Microdiseño

**CASO DE USO: Administrar Documento de Tesis**

Versión 1.0

Historial de Versiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 16/10/2014 | 1.0 | Versión preliminar como una propuesta de desarrollo. | André Calderón |

|  |  |
| --- | --- |
| **Detalle de Diseño N°** | 01 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Elaborado Por:** | **ANDRÉ CALDERÓN** | | | |
| **Descripción General:** | Este documento está orientado a servir de guía en el desarrollo de la funcionalidad:    ***Administrar Documento de Tesis:***     * Esta funcionalidad tiene como finalidad de registrar Título de la tesis, autor, fecha de publicación, tema, especialidad, código de la tesis, estado de la tesis, docente asesor, grado académico de documentos de tesis. * Además, de permitir a los usuarios visualizar y eliminar el documento. | | | |
| **Referencias:** | **Requerimientos:** | SisCoTe\_ECU\_ADT.docx  Caso de Uso # 1 | | |
| **Diseño de Interfaz Usuario:** | SisCoTe\_ECU\_ADT.docx  Prototipos asociados al Caso de Uso # 1 | | |
| **Diseño de**  **Arquitectura:** | SisCote\_DD.docx | | |
| **Otros:** | Ninguno | | |
| **Clases**  **Requeridas:** | **Nombre de la Clase** | | **Stereotype** | **Acción** |
| DAOTesisImpl | | DAO | Crear |
| DAOTesis.java | | DAO Interface | Crear |
| Documento.java | | Domain Class | Crear |
| AdministrarTesis.java | | Servicie | Crear |
| **Clases Java Swing** | **Nombre del archivo de vista (JSF)** | | | **Acción** |
| UIAdministración.java | | | Crear |
| **Descriptores Hibernate - XML** | **Nombre del archivo descriptor Hibernate - XML** | | | **Acción** |
| hibernate.cfg.xml | | | Crear |
| Hibernate.reveng.xml | | | Crear |
| **Tablas:** | **Nombre de Tabla** | | **Tipo de Objeto** | |
| CATEGORIA (select) | | Tabla | |
| DOCUMENTO (select, insert, update) | | Tabla | |
| TEMA (select, insert) | | Tabla | |
| AUTOR (select, insert, update) | | Tabla | |
|  | ESPECIALIDAD (select) | | Tabla | |

|  |  |
| --- | --- |
| **Detalle de Implementación a Nivel de Clases:** | **pe.com.siscote.model.Documento**     * Clase java utilizada para mapear y tener acceso la información de la tabla Documento.      * **Métodos**:     Dado que se trata de una clase de dominio que solo será utilizado como un objeto consultador de información, los únicos métodos a implementar son los accesores: ***Getters* y *Setters*** por cada propiedad/atributo de la clase.     * 1. **public TipoDatosAtributo** getNombreAtributo()   2. **public void** setNombreAtributo()      * **Acción:**     Crear |
| **Detalle de implementación a nivel de Servicios** | **pe.com.siscote.model.service.AdministrarTesis**       * Clase que contiene la lógica de negocio del sistema, con las operaciones de de agregar, modificar, eliminar y buscar documentos de tesis.      * **Métodos**:      * 1. **public void** AgregarTesis()   2. **public Boolean** EliminarTesis()   3. **public Boolean** ModificarInfoTeis()   4. **public Boolean** ModificarDocTeis()   5. **public Documento** BuscarTesis(Documento)      * **Acción:**     Crear |
| **Detalle de implementación a nivel de DAOs** | **pe.com.sirh.model.dao.DAOTesis**     * Interfase utilizada para desacoplar las capas implementadas en el sistema (capa de *Datos* de la capa *Negocio*).      * **Métodos:**      * 1. **public void** crearDocumento(Documento)   2. **public void** modificarDocumento(Documento)   3. **public Documento** consultarDocumento(**INT** idDocumento) 4. **public List**<Documento> listarDocumento()      * **Acción:**     Crear    **pe.com.sirh.model.dao.impl.DAOTesisImpl**     * Clase de acceso a datos que extiende de la clase Hibernate ***y HibernateUtil*** e implementa la interfase ***DAOTesis***. Implementa métodos de acceso a los las clases de dominio descritas en el hibernate.cfg.xml.      * **Métodos:**      * 1. **public void** crearDocumento(Documento)   2. **public void** modificarDocumento(Documento)   3. **public Documento** consultarDocumento(**INT** idDocumento) 4. **public List**<Documento> listarDocumento()        * **Acción:**     Crear |
| **Detalle de Implementación a Nivel de JSF:** | **UIAdministración.java**   * **Acción:**     Crear     * **Descripción:**   Clase swing de la vista que ofrece la funcionalidad de administrar los documentos de tesis, permite registrarlos, modificarlos o buscar.    Contiene los campos en un formulario:    Nombres del autor  Título del documento  Tema del documento  Año de publicación del documento  Descripción del documento |
| **Detalle de Implementación a Nivel de Tablas:** | **Tabla DOCUMENTO**    Tabla que almacena la información del documento:    CREATE TABLE IF NOT EXISTS `mydb`.`Documento` (  `idDocumento` INT UNSIGNED NOT NULL AUTO\_INCREMENT ,  `Codigo` VARCHAR(45) NOT NULL ,  `Titulo` VARCHAR(45) NOT NULL ,  `Estado` TINYINT(1) NOT NULL ,  `Fecha de Publicacion` DATETIME NOT NULL ,  `Categoria\_idCategoria` INT UNSIGNED NOT NULL ,  `Especialidad\_idEspecialidad` INT UNSIGNED NOT NULL ,  PRIMARY KEY (`idDocumento`) ,  UNIQUE INDEX `idDocumento\_UNIQUE` (`idDocumento` ASC) ,  UNIQUE INDEX `Codigo\_UNIQUE` (`Codigo` ASC) ,  UNIQUE INDEX `Titulo\_UNIQUE` (`Titulo` ASC) ,  INDEX `fk\_Documento\_Categoria1\_idx` (`Categoria\_idCategoria` ASC) ,  INDEX `fk\_Documento\_Especialidad1\_idx` (`Especialidad\_idEspecialidad` ASC) ,  CONSTRAINT `fk\_Documento\_Categoria1`  FOREIGN KEY (`Categoria\_idCategoria` )  REFERENCES `mydb`.`Categoria` (`idCategoria` )  ON DELETE NO ACTION  ON UPDATE NO ACTION,  CONSTRAINT `fk\_Documento\_Especialidad1`  FOREIGN KEY (`Especialidad\_idEspecialidad` )  REFERENCES `mydb`.`Especialidad` (`idEspecialidad` )  ON DELETE NO ACTION  ON UPDATE NO ACTION)  ENGINE = InnoDB |