Universidad Nacional de Misiones

Facultad de Ciencias Exactas, Químicas y Naturales

Carrera

Licenciatura en Sistemas de Información

Cátedra

Programación Orientada a Objetos I

Titular: Lic. Claudio O. Biale

Nombre del Trabajo:

Trabajo Integrador

v0.1

Alumno:

Ulises C. Ramirez

Legajo Universitario: LS00704

Año: 2017

Índice de Contenidos

Seguimiento de entregas....3
1. Proyecto 1....4
2. Objetivos del Sistema....5
  2.1 Seguimiento de Reparaciones:....5
  2.2 Manejo de Reclamos:....5
3. Modelo de Dominio....6
4. Diseño....7
  4.1 Diagrama de Clases....7
    4.1.1 Identificación de las Clases....7
    4.1.2 Identificación de los Atributos....8
    4.1.3 Identificación de los Métodos....9
    4.1.4 Primer versión del Diagrama de Clases (Solo Atributos)....11
Anexos....12
  Anexo 1: Software Utilizado....12
  Anexo 2: Bibliografía....12

Seguimiento de entregas

Versión	Contenido	Modificaciones con respecto a versión anterior
0.1	1. Objetivos 2. Modelo de Dominio 3. Identificación de Clases, Atributos y Métodos 4. Diagrama de Clases (v1.0)	---

Trabajo Integrador
Titular: Lic. Claudio O. Biale

1. Proyecto 1

Una empresa de reparación de artículos, por ejemplo, hardware de computadoras desea implementar un sistema de gestión de reclamos.

El sistema debe soportar el ingreso de reclamos, la asignación de recursos para la reparación de los artículos y el seguimiento de las reparaciones.

El sistema deberá registrar para cada reclamo un número, la descripción del problema, el tipo del artículo a reparar, la fecha en la que fue ingresado al sistema y la fecha estimada de entrega. De los tipos de artículos interesa registrar un código (que lo identifica) y un nombre.

Los reclamos serán reparados por los técnicos de la empresa los cuales deberán estar registrados en el sistema. De cada técnico interesa saber su documento único, nombres, apellidos y los tipos de artículo sobre los que está capacitado para trabajar. Un técnico puede trabajar como empleado mensual o jornalero. En caso de ser jornalero interesa conocer la tarifa por hora que se le paga, mientras que para un empleado mensual interesa conocer el sueldo mensual que percibe.

Para realizar una reparación, se planifica de entre el conjunto de tareas definidas en el sistema, una secuencia de tareas que son las que los técnicos deberán de realizar para concluir la reparación. De las tareas interesa su código único, nombre, descripción y tipos de artículos a los que aplica. Cada tarea de una reparación será efectuada por un técnico de la empresa capacitado en el tipo de artículo a reparar.

Es de interés para la empresa llevar un registro del tiempo invertido por el técnico en cada tarea realizada. Por ello, será necesario registrar en el sistema la cantidad de horas dedicadas y el día que se realizó la tarea. Si la misma es efectuada en más de un día, interesa saber cuántas horas le dedicó para cada fecha. Se debe indicar cuando se finaliza la tarea.

2. Objetivos del Sistema

Teniendo en cuenta las necesidades planteadas en el enunciado se proponen los dos siguientes objetivos principales a los cuales la implementación de este sistema debe dar solvencia.

2.1 Seguimiento de Reparaciones:

El sistema deberá ser capaz de acompañar al usuario mediante el registro y actualización constante de las diferentes instancias por las cuales pueda pasar un reclamo, llevando el registro del estado del mismo en todo momento.

2.2 Manejo de Reclamos:

El sistema deberá ser capaz de permitir el ingreso de diferentes reclamos que harán referencia a diferentes artículos, estos reclamos deberán ser atendidos mediante un cronograma de reparaciones.

3. Modelo de Dominio

El Análisis del escenario propuesto por la cátedra resultó en el siguiente Modelo de Dominio, cuyo propósito es brindar una perspectiva clara y amplia del problema.

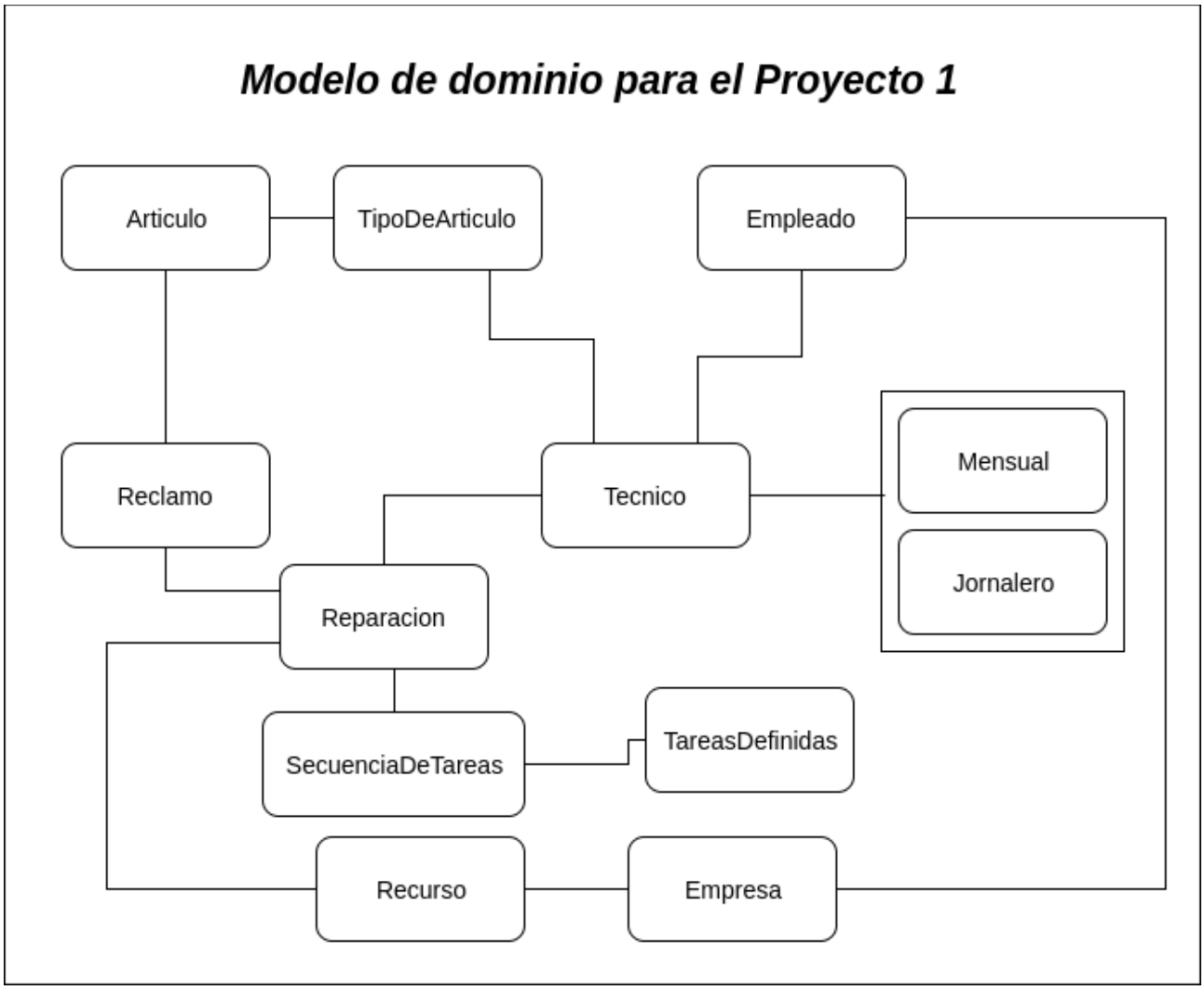

Diagrama 1: Modelo de dominio para el Proyecto 1

A partir del Modelo de Dominio (Diagrama 1) se puede empezar a bosquejar las clases que conformarán el diagrama de Clases.

4. Diseño

4.1 Diagrama de Clases

4.1.1 Identificación de las Clases

Para esto, se puede hacer uso de la guía[1] brindada por la cátedra en donde como primer paso podemos ocuparnos en tratar de identificar las clases que tendremos en el modelo, ademas de las directivas que encontramos en la guía mencionada anteriormente podemos apoyarnos en el modelo de dominio que se ha confeccionado. A continuación se presentan las clases identificadas:

Diagrama 2: Clases identificadas

Aquí se puede apreciar las clases que se identificaron del enunciado para el Proyecto Numero 1, a continuación se procederá a la justificación de las clases mencionadas anteriormente.

Clases/Interfaces	Justificación
`interface AltaBajaModificacion`	Dado a que vamos a necesitar la gestión de varias entidades dentro del sistema, entendiendo como 'Gestión' la habilidad de que se realicen acciones de tipo Alta, Baja, Modificación y Consulta se establece una interfaz que actuara como molde y obligará a las entidades que la implementen que definan estos métodos que son necesarios para la manipulación de las instancias de esas clases.
`abstract class Empleado implements AltaBajaModificacion`	Superclase que permitirá añadir un nivel de abstracción adicional, esto nos permite flexibilizar el modelo, ya que por ejemplo, podríamos seguir agregando roles a de empleados sin mucha dificultad. Ademas esto nos ayuda a tener un lienzo para poder ubicar las diferentes acciones comunes que sea posible realizar por las diferentes clases que heredan de esta.
`class Empresa implements AltaBajaModificacion`	Esta entidad permite que el modelo acepte varias Empresas, lo cual da un gran potencial de crecimiento a la organización si que esta se vea restricta por alguna decisión de esta índole en el diseño del modelo.
`class Tecnico extends Empresa`	Esta entidad representa el rol de un técnico que pertenece a la empresa, esta decisión de modelo permite identificar y controlar las acciones que tienen permitidas realizar los tecnicos que se encuentren registrados en el sistema ademas de permitir llevar registrada información asociada al tecnico
`class Operador extends Empresa`	Al igual que la entidad Técnico esta clase nace para representar el

1 Anexo 2: [COB2017b] Modelo.pdf

	rol de un Operador dentro de la organización, entendiendo como 'Operador' a la persona que se encarga de recibir los reclamos.
class TipoDeArticulo	Esta entidad surge con la necesidad de representar un atributo de la clase Artículo, es necesario para que el sistema sea mas parametrizable y como consecuencia a esto hacer la implementación del sistema mucho mas flexible y capaz de adaptarse a entornos con requerimientos diferentes a este.
class Articulo	Esta representa a un articulo que va a ingresar al sistema con propósito de ser reparado.
Class Reparacion	Esta entidad representa a la cabecera del C**onjunto de Tareas** a desarrollar para que un **Reclamo** sea solucionado.
Class Recurso	Esta clase representa a un recurso que posee la **Empresa**, este recurso podrá ser asignado a una **Reparacion** con el fin de que un **Reclamo** sea solucionado.
Class SecuenciaDeTareas	Esta clase representara un conjunto de objetos asociados a una Reparación, en donde cada objeto representa una tarea que debe cumplirse para que el la **Reparacion** y por ende el **Reclamo** estén cumplidos.
Class TareasDefinidas	Esta clase representa las **Tareas** que se pueden realizar a un **Articulo** con un **TipoDeArticulo** determinado, nuevamente esta clase surge con el fin de hacer al sistema mas parametrizable
Class Reclamo	Esta es una entidad que se crea dar de alta un nuevo **Reclamo** en el sistema, a esta se le asociará una instancia de **Reparacion** una vez se este listo para atender el reclamo

4.1.2 Identificación de los Atributos

Clases/Interfaces	Atributos	Observaciones
interface AltaBajaModificacion		
abstract class Empleado implements AltaBajaModificacion	1. nombres:String 2. apellidos:String 3. numDocumento:String	*'[...]De cada técnico interesa saber su documento único, nombres, apellidos y los tipos de artículo sobre los que está capacitado para trabajar. Un técnico puede trabajar como empleado mensual o jornalero[...]', aquí se decidió poner estos atributos en una superclase ya que estos datos serian compartidos por cualquier empleado de la empresa, incluso hubiera sido bueno incluir ademas una superclase denominada Persona, y de esta manera si la empresa en un futuro quisiera registrar clientes no haría falta cambios.*
class Empresa implements AltaBajaModificacion	1. id:int 2. nombre:String 3. cuit:String	
class Tecnico extends Empresa	1. tiposArticuloCapacitado:Tipo DeArticulo[]	Lo que se quiere expresar acá es un atributo con el cual se pueda definir la colección de artículos en los cuales el técnico está capacitado a trabajar
class Operador extends Empresa		
class TipoDeArticulo	1. codigo:String 2. nombre:String	
class Articulo	1. nombre:String 2. tipoArticulo:TipoDeArticulo	
Class Reparacion	1. RecursosAsignados:Recursos[]	

Class Recurso	1. nombre:String 2. canDisponible:int	
Class SecuenciaDeTareas	1. tarea:TareasDefinidas 2. fechaInicio:Date 3. fechaFin:Date	Aqui se quiere expresar que las tareas necesarias para la reparacion van a estar compuestas por una lista de tareas.
Class TareasDefinidas	1. codigoUnico:String 2. nombre:String 3. descripcion:String 4. tipoArticulos:TipoDeArticulo[]	
Class Reclamo	1. numero:int 2. descProblema: String 3. articulo:Articulo 4. fechaEntrada:Date 5. fechaEstimEntrega:date	

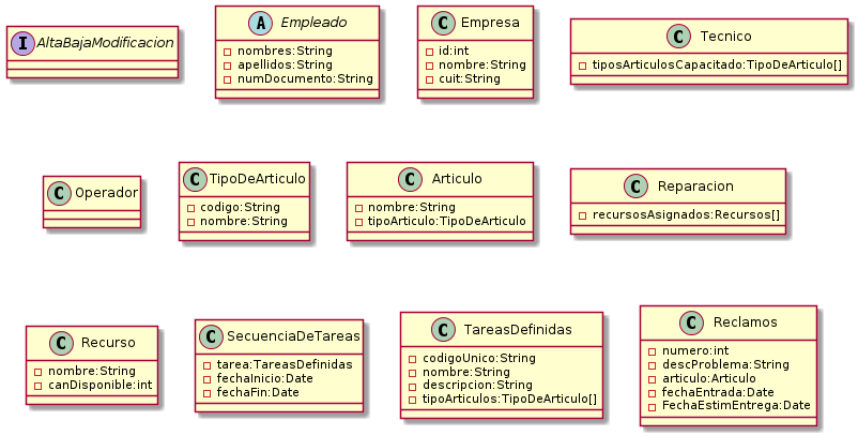

Diagrama 3: Clases identificadas junto con los atributos identificados

4.1.3 Identificación de los Métodos

Nuevamente en este apartado del documento se hará uso del documento facilitado por la cátedra, en la siguiente tabla tendremos las clases identificadas con sus respectivos métodos identificados.

Cabe destacar que los métodos **getters,** los métodos **setters** y los constructores no se incluirán en la tabla, pero estos estarán presentes.

interface AltaBajaModificacion	1. Alta(Objeto:Object) : Se encargará de dar de alta a una nueva entidad en el sistema. 2. Baja(Objeto:Object) : Se encargará de dar de baja a una entidad del sistema 3. Modificacion(Objeto:Object) : Permitirá realizar cambios a las entidades que existen en el sistema 4. Consulta(Objeto:Object) : Permitirá obtener las diferentes instancias de una entidad en el sistema.
abstract class Empleado implements AltaBajaModificacion	1.
class Empresa implements	1. Alta()

AltaBajaModificacion	2. Baja() 3. Modificacion() 4. Consulta()
class Tecnico extends Empleado implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
class Operador extends Empleado implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
class TipoDeArticulo implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
class Articulo implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
Class Reparacion implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta() 5. AsignarRecurso(unRecurso: Recurso, Cant: int): esta tarea permite asignarle una cierta cantidad de un recurso a la reparación para que asi esta sea solucionada
Class Recurso implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
Class SecuenciaDeTareas	1. Alta() 2. Baja() 3. SolucionarProblema(): La responsabilidad de este método es marcar el estado de la tarea como lista. 4. AgregarTareaDefinida(): La responsabilidad de este metodo es el de agregar una nueva tarea a la cola de tareas por realizar para que la **Reparacion** y por ende el **Reclamo** sean marcado como solucionados 5. QuitarTareaDefinida(): La responsabilidad de este método es la de quitar una tarea de la cola de tareas.
Class TareasDefinidas implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()
Class Reclamo implements AltaBajaModificacion	1. Alta() 2. Baja() 3. Modificacion() 4. Consulta()

4.1.4 Primer versión del Diagrama de Clases (Solo Atributos)

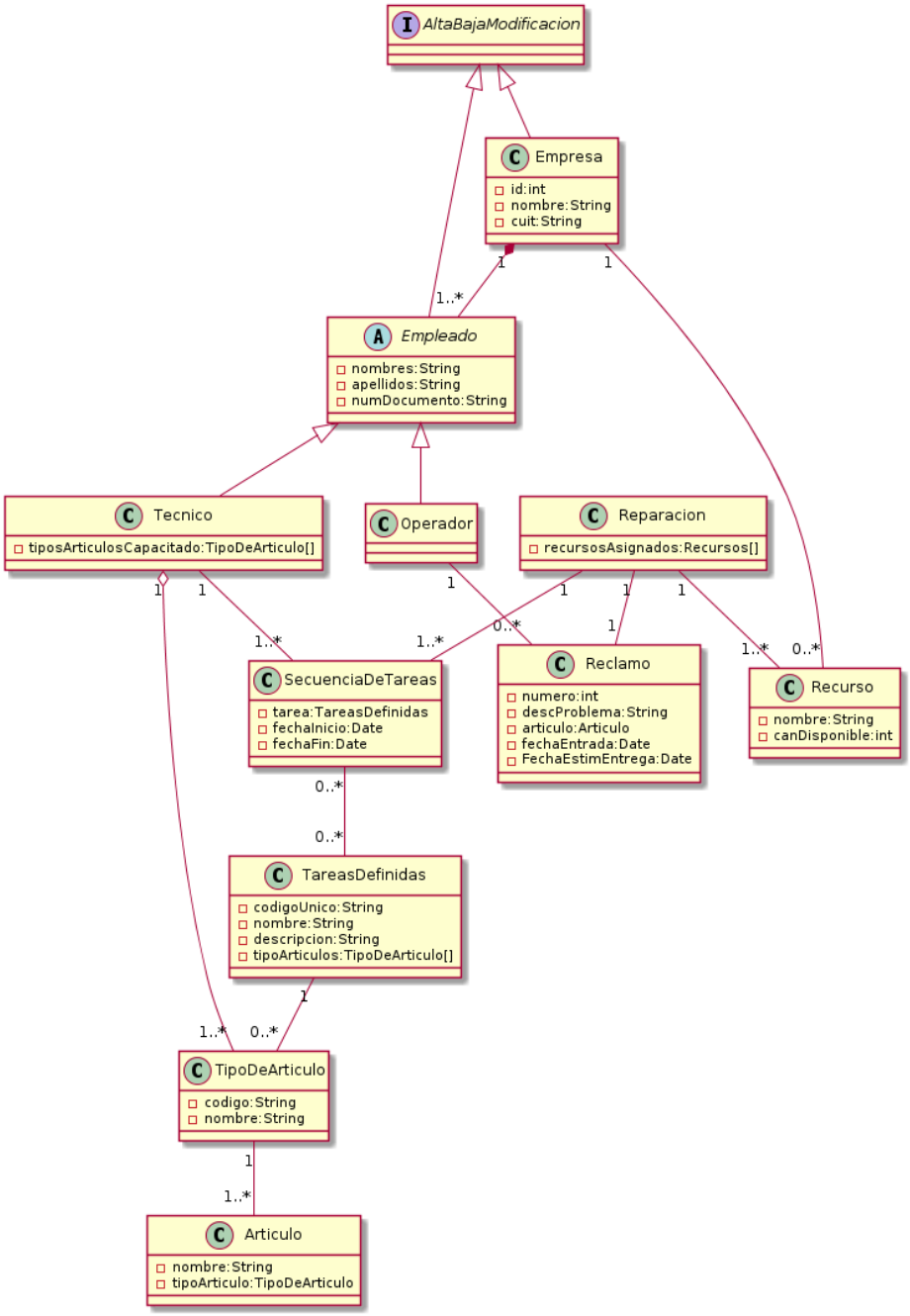

Diagrama 4: Primer versión del Diagrama de Clases, únicamente con los atributos

Anexos

Anexo 1: Software Utilizado

Documento de Texto	LibreOffice Writer v5.1.6.2
Diagramas de Casos de Uso	Google Draw.io PlantUML
Modelo de Dominio	
Diagramas de Clase	
Diagramas de Secuencia de Sistemas	
Diagramas de Secuencia de Diseño	
Implementación	NetBeans v8.2

Anexo 2: Bibliografía

1. [COB2017a] Lic. Claudio O. Biale, *Diapositivas brindadas por la cátedra*, Universidad Nacional de Misiones, Facultad de Ciencias Exactas Químicas y Naturales.
2. [COB2017b] Lic. Claudio O Biale, *Modelo*, Universidad Nacional de Misiones, Facultad de Ciencias Exactas Químicas y Naturales.
3. [KEN11] Kenneth E. Kendall, Julie E. Kendall, *Análisis y Diseño de Sistemas, Octava Edición*, Editorial Pearson Educación
4. [CRAIG04] Craig Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Tercera Edición, Editorial Addison Wesley Professional