Programación 4

Laboratorio 5

Grupo X
Integrantes

Giuliano Kuns **CI:** 4.615.666-7
Rodrigo Negreyra **CI:** 5.098.239-5
Ruben Silveira **CI:** 5.158.681-1
José Antonio Esteche Ferreira **CI:** 4.929.962-2

Docente: Carmela Beiro

Introducción

1.1 Propósito

El propósito de este documento es brindar una descripción general del Modelo de Diseño.

1.2 Alcance

El informe del Modelo de Diseño presenta una abstracción de la solución lógica al problema. Incluye las colaboraciones que realizan cada uno de los casos de uso del Modelo de Casos de Uso.

1.3 Estructura del Documento

El documento está dividido en tres secciones. La segunda sección presenta la organización lógica del sistema en paquetes de diseño. La tercera sección presenta las colaboraciones con la realización de los casos de uso incluidos en el Modelo de Casos de Uso. Por último, la cuarta sección presenta los criterios generales adoptados para el diseño de las colaboraciones.

A lo largo de esta plantilla se encuentran comentarios y ejemplos acerca del contenido del documento a elaborar a partir de ésta. Éstos se encuentran indicados, al igual que este párrafo, por una línea a lo largo de su borde izquierdo. Estas secciones deben ser eliminadas de la versión final del documento.

2 Realización de Casos de Uso

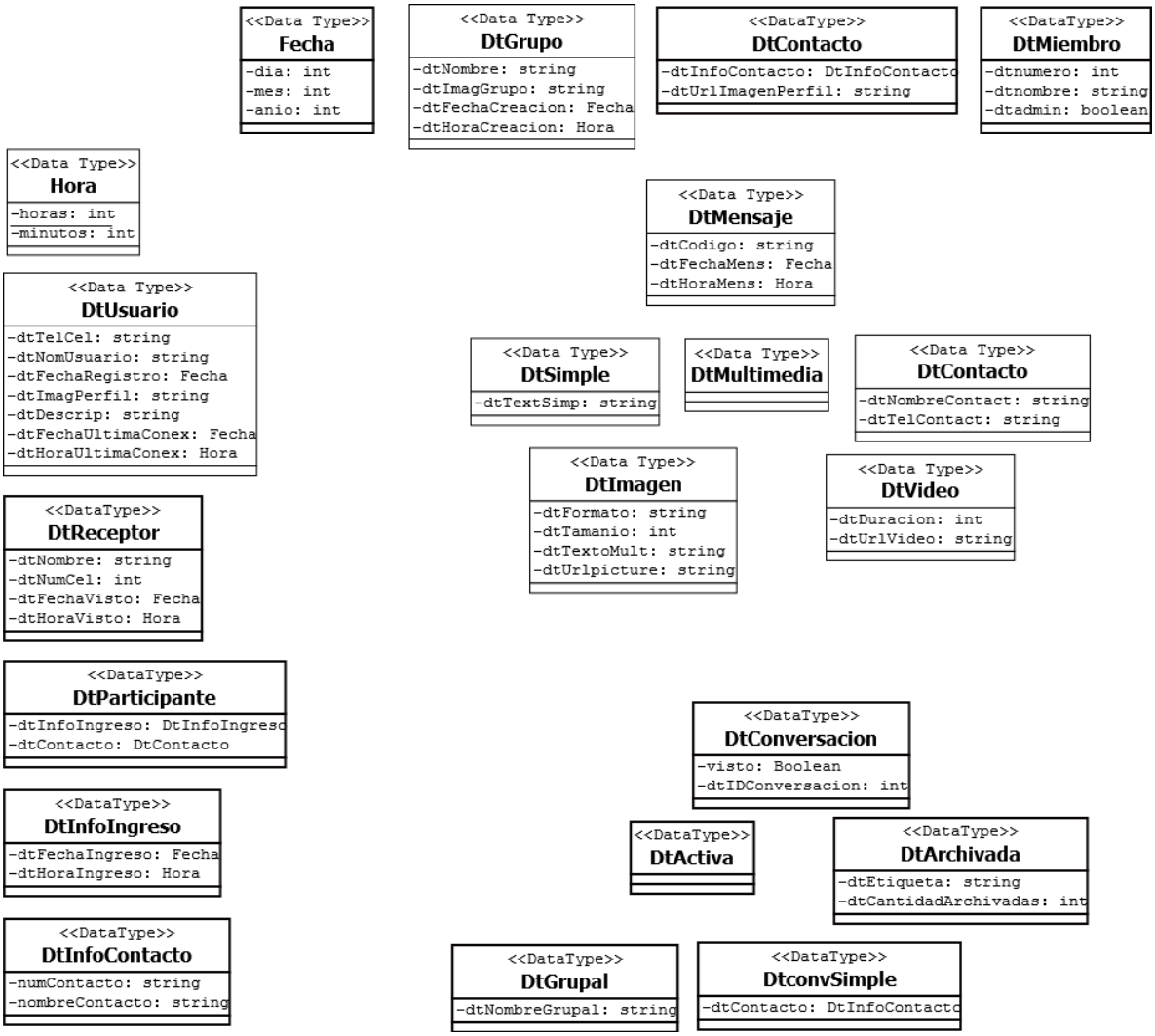
Las realizaciones, están expresadas en términos de estructura e interacciones entre instancias que respeten dicha estructura. Concretamente, la parte estructural de una realización es un *diagrama* de clases conteniendo clases del modelo; la parte dinámica es un conjunto de *diagramas* de interacción que ilustran el flujo de mensajes entre instancias de las clases de la parte estructural correspondiente.

Esta sección está dividida por colaboración. Una colaboración es la realización de uno o más casos de uso. Para cada colaboración habrá una sección que muestre su diseño. El nombre de una colaboración corresponde al nombre del caso de uso que realiza, o el nombre del "área temática" determinada por el conjunto de casos de uso que realiza. La estructura para presentar la información será la siguiente:

Estructura

En esta sección se presenta el diagrama de clases correspondiente al diseño de ésta colaboración.

Data Types:



Clases:

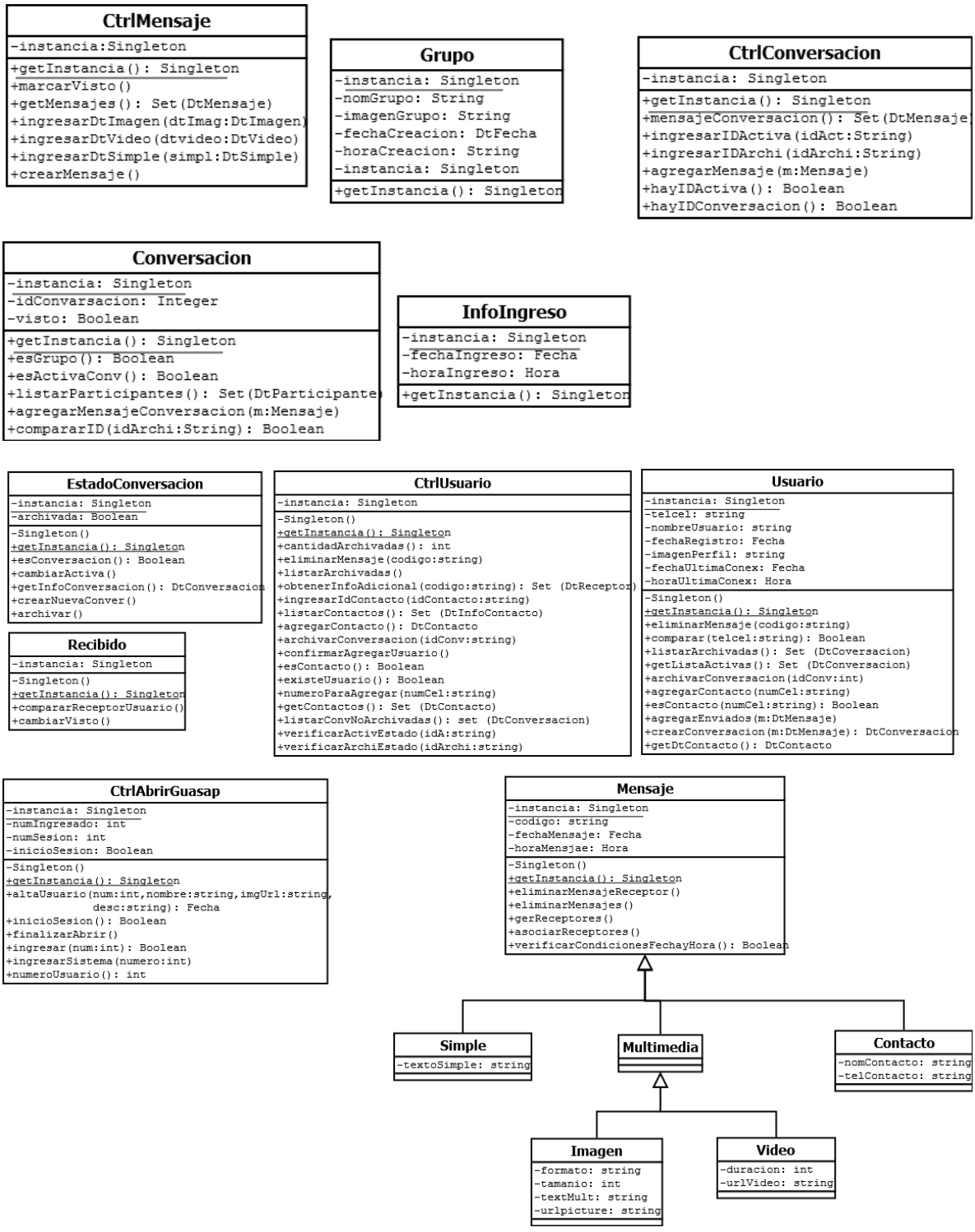
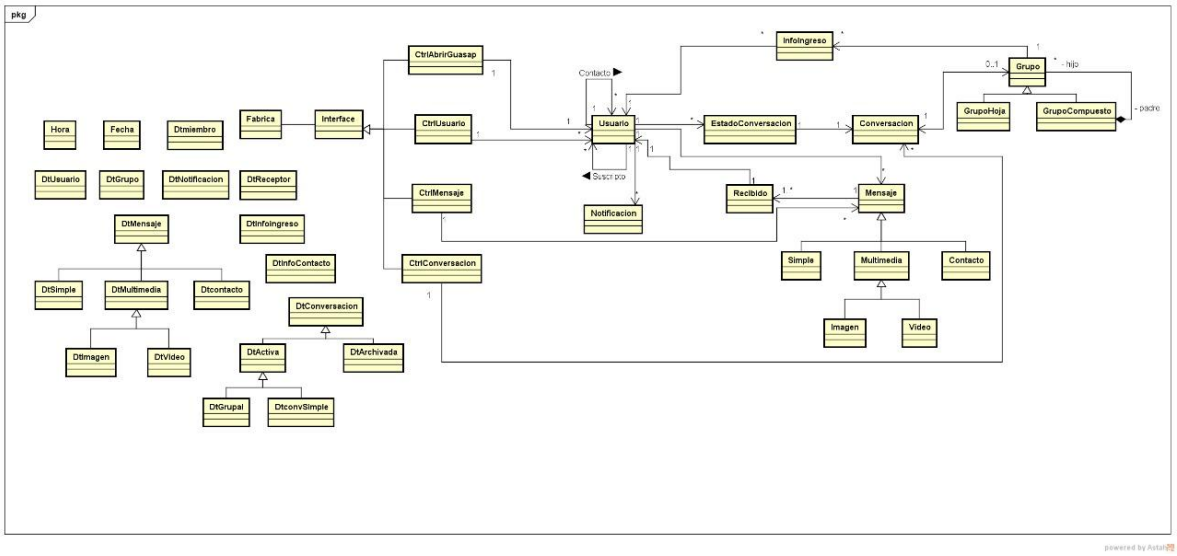

CtrlMensaje
-instancia:Singleton
+getInstancia(): Singleton
+marcarVisto()
+getMensajes(): Set(DtMensaje)
+ingresarDtImagen(dtImag:DtImagen)
+ingresarDtVideo(dtvideo:DtVideo)
+ingresarDtSimple(simpl:DtSimple)
+crearMensaje()
Grupo
-instancia: Singleton
-nomGrupo: String
-imagenGrupo: String
-fechaCreacion: DtFecha
-horaCreacion: String
-instancia: Singleton
+getInstancia(): Singleton
CtrlConversacion
-instancia: Singleton
+getInstancia(): Singleton
+mensajeConversacion(): Set(DtMensaje)
+ingresarIDActiva(idAct:String)
+ingresarIDArchi(idArchi:String)
+agregarMensaje(m:Mensaje)
+hayIDActiva(): Boolean
+hayIDConversacion(): Boolean
Conversacion
-instancia: Singleton
-idConvarsacion: Integer
-visto: Boolean
+getInstancia(): Singleton
+esGrupo(): Boolean
+esActivaConv(): Boolean
+listarParticipantes(): Set(DtParticipante)
+agregarMensajeConversacion(m:Mensaje)
+compararID(idArchi:String): Boolean
InfoIngreso
-instancia: Singleton
-fechaIngreso: Fecha
-horaIngreso: Hora
+getInstancia(): Singleton
EstadoConversacion
-instancia: Singleton
-archivada: Boolean
-Singleton()
+getInstancia(): Singleton
+esConversacion(): Boolean
+cambiarActiva()
+getInfoConversacion(): DtConversacion
+crearNuevaConver()
+archivar()
Recibido
-instancia: Singleton
-Singleton()
+getInstancia(): Singleton
+compararReceptorUsuario()
+cambiarVisto()
CtrlUsuario
-instancia: Singleton
-Singleton()
+getInstancia(): Singleton
+cantidadArchivadas(): int
+eliminarMensaje(codigo:string)
+listarArchivadas()
+obtenerInfoAdicional(codigo:string): Set (DtReceptor)
+ingresarIdContacto(idContacto:string)
+listarContactos(): Set (DtInfoContacto)
+agregarContacto(): DtContacto
+archivarConversacion(idConv:string)
+confirmarAgregarUsuario()
+esContacto(): Boolean
+existeUsuario(): Boolean
+numeroParaAgregar(numCel:string)
+getContactos(): Set (DtContacto)
+listarConvNoArchivadas(): set (DtConversacion)
+verificarActivEstado(idA:string)
+verificarArchiEstado(idArchi:string)
Usuario
-instancia: Singleton
-telcel: string
-nombreUsuario: string
-fechaRegistro: Fecha
-imagenPerfil: string
-fechaUltimaConex: Fecha
-horaUltimaConex: Hora
-Singleton()
+getInstancia(): Singleton
+eliminarMensaje(codigo:string)
+comparar(telcel:string): Boolean
+listarArchivadas(): Set (DtCoversacion)
+getListaActivas(): Set (DtConversacion)
+archivarConversacion(idConv:int)
+agregarContacto(numCel:string)
+esContacto(numCel:string): Boolean
+agregarEnviados(m:DtMensaje)
+crearConversacion(m:DtMensaje): DtConversacion
+getDtContacto(): DtContacto
CtrlAbrirGuasap
-instancia: Singleton
-numIngresado: int
-numSesion: int
-inicioSesion: Boolean
-Singleton()
+getInstancia(): Singleton
+altaUsuario(num:int,nombre:string,imgUrl:string,
desc:string): Fecha
+inicioSesion(): Boolean
+finalizarAbrir()
+ingresar(num:int): Boolean
+ingresarSistema(numero:int)
+numeroUsuario(): int
Mensaje
-instancia: Singleton
-codigo: string
-fechaMensaje: Fecha
-horaMensjae: Hora
-Singleton()
+getInstancia(): Singleton
+eliminarMensajeReceptor()
+eliminarMensajes()
+gerReceptores()
+asociarReceptores()
+verificarCondicionesFechayHora(): Boolean
Simple
-textoSimple: string
Multimedia
Contacto
-nomContacto: string
-telContacto: string
Imagen
-formato: string
-tamanio: int
-textMult: string
-urlpicture: string
Video
-duracion: int
-urlVideo: string

Diagrama:



2.1.1 Design Patterns

En esta sección se indican todos los Design Patterns involucrados en el diseño de la colaboración de esta sección. En caso de que la aplicación de este Design Pattern ya haya sido documentada (esto implica que las clases involucradas en el patrón sean las mismas) en alguna sección anterior sólo debe hacerse referencia a la misma. En caso que no se haya aplicado ningún Design Pattern esta sección no debe incluirse.

Factory:

Se utilizo un patrón de Factory para desacoplar la capa de presentación con los controladores.

Las clases involucradas son:

- la capa de presentación, que cumple el rol de consumidor.
- Fabrica, que cumple el rol de Fabrica.
- Interface, que cumple el rol de Provedor.
- Todos los controladores, que cumplen el rol de Provedores Concretos.

Singleton:

Se utilizo un patrón de Singleton para todas las clases, para asegurarnos de tener una única instancia para cada una.

Las clases involucradas son:

- Todas menos los datatypes

Composite:

Se utilizo un patrón de Composite para la clase grupo, con el fin de solucionar el problema de jerarquías de grupos

Las clases involucradas son:

- Grupo, que cumple el rol de Componente.
- GrupoHoja, que cumple el rol de Hoja.
- GrupoCompuesto, que cumple el rol de Compuesto.

Observer:

Se utilizo un patrón de Observer para la clase usuario, con el fin de poder mandar notificaciones a los usuarios subscriptos.

Las clases involucradas son:

-En este caso la clase usuario es Subject, Observer y Observer Concreto