Descripcion diseño

En términos generales, los requerimientos funcionales son traducibles en manejar (crear, acceder, editar y mantener) tipos abstractos de datos. Cuando creo una reserva por ejemplo, únicamente debo verificar disponibilidad de vehículos, crear una instancia de una clase reserva y actualizar el estado del vehículo, por ejemplo. Es por esto que, siguiendo esta aproximación, creemos pertinente utilizar un modelo de control centralizado, que nos permite dividir fácilmente el trabajo entre los miembros del grupo y simplificar el problema respetando encapsulamiento. En esa medida, proponemos una división primaria en tres partes. En primera instancia, un *modelo* compuesto solo de *data Holders* con únicamente getters y setters representando los tipos abstractos de datos que se deben manejar (ej carro y cede). En segunda instancia, un *controller* compuesto en principio de 4 clases, representando cada una a un usuario (admin, cliente, empleado o admin general) con estereotipo de *controllers*, en el sentido de que deben definir como manejar los datos según lo que quiera hacer cada usuario. Además, en este nivel se incluye una clase *sistema empresa* como base de datos general, y se maneja la persistencia con una clase download, que carga los datos de los archivos y los guarda en la base de datos y otra upload, que reescribe los archivos actualizandolos cada vez que un usuario cierra sesión. Estas 4 clases de usuario con sus respectivas responsabilidades nos las dividimos entre nosotros. (nótese que si alguno considera pertinente agregar nuevas clases en este nivel para manejar lógica compleja acorde a sus requerimientos, lo puede hacer sin problema).Por último, el programa contará con una consola, que regula la interacción con los usuarios mediante login y menús de acciones.