

### TECNOLÓGICO NACIONAL DE MÉXICO



#### INSTITUTO TECNOLÓGICO DEL VALLE DE OAXACA

DEPARTAMENTO DE: CIENCIAS ECONÓMICO - ADMINISTRATIVA

PROGRAMA EDUCATIVO: INGENIERÍA INFORMÁTICA

ASIGNATURA: DESARROLLO BACK-END

UNIDAD 3. PATRONES DE COMPORTAMIENTO

**ACTIVIDAD 3.2 DIAGRAMA DE CLASES** 

DOCENTE: CARDOSO JIMÉNEZ AMBROSIO

ESTUDIANTE: ESTHER MEJÍA ANTONIO

OCTAVO SEMESTRE GRUPO: 18A

CICLO: ENERO - JUNIO DE 2023

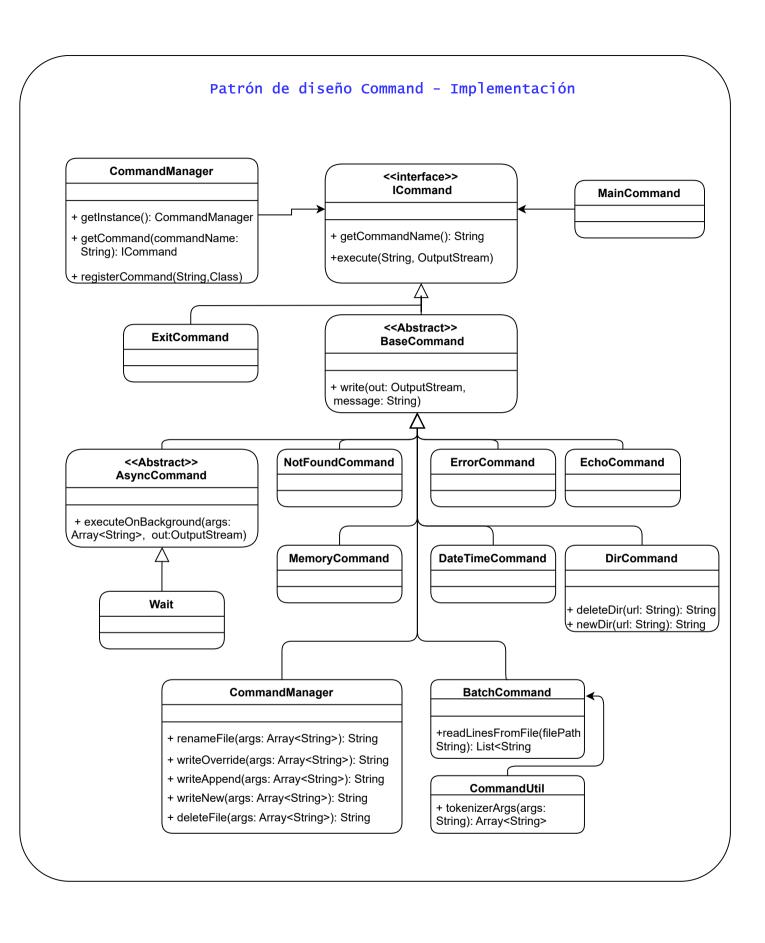
EX HACIENDA DE NAZARENO, XOXOCOTLÁN, OAXACA.

02 DE JUNIO DEL 2023.

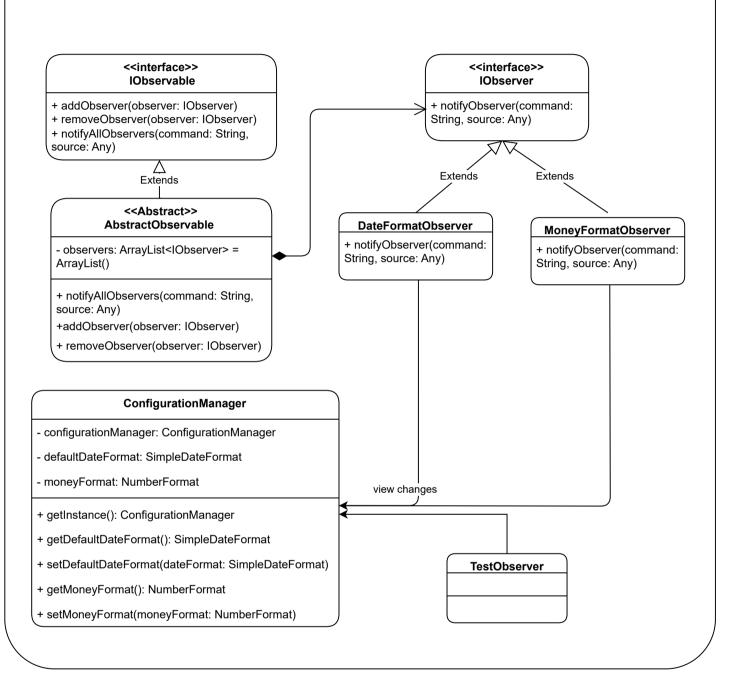
### **CONTENIDO**

- 3.1 Iterator
- 3.2 Command
- 3.3 Observer
- 3.4 Template method
- 3.5 Strategy
- 3.6 Chain of Responsability
- 3.7 Interpreter
- 3.8 Mediator
- 3.9 Memento
- 3.10 Null object
- 3.11 State
- 3.12 Visitor

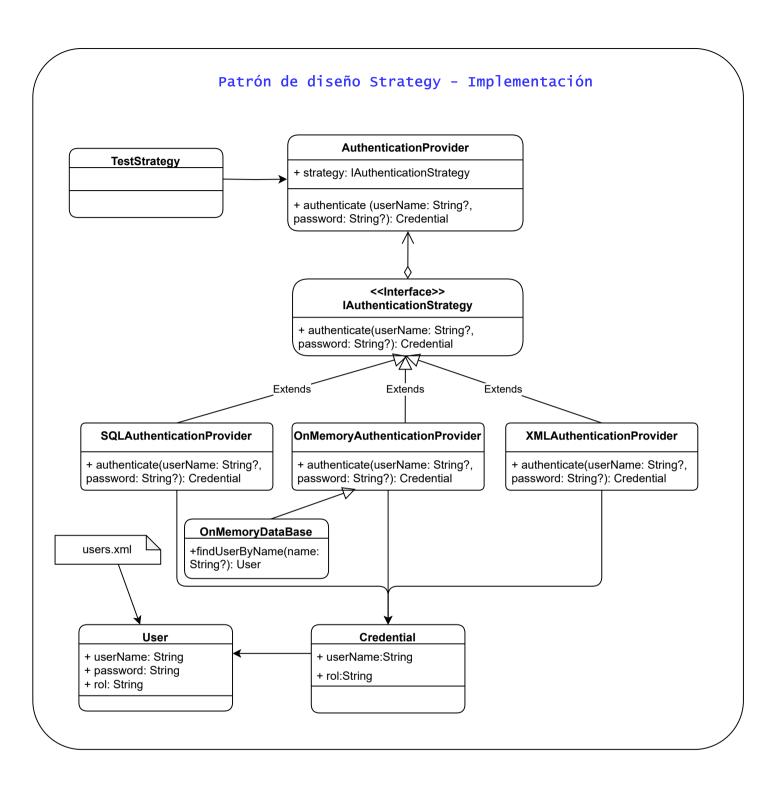
### Patrón de diseño Iterator - Implementación <<interface>> Testiterator <<interface>> IContainer<T> Ilterator + getIterator():Iterator <---use----· - - -use- - -> + hasNext(): Boolean + next(): T Extends Extends TreeEmployeeIterator **Employee** + hasNext(): Boolean + next(): Employee + createIterator(): IIterator<Employee> -Pertenece + getSubordinates(): List<Employee> + setSubordinates(subordinates: MutableList<Employee> + addSubordinate(subordinate: Employee?) + toString(): String



### Patrón de diseño Observer - Implementación

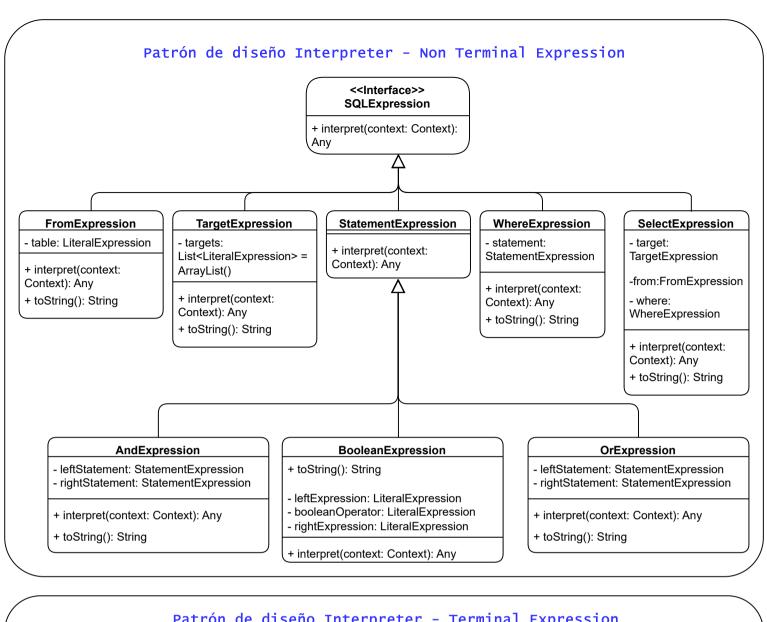


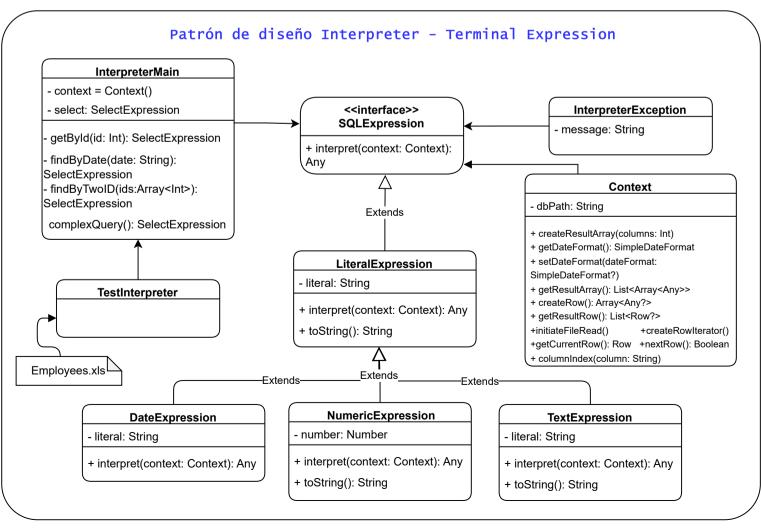
# Patrón de diseño TemplateMethod - Implementación **TemplateFileProcess TestTemplateMethod** + file: File + logPath: String + movePath: String + execute() # validateName() # validateProcess() # processFile() # createLog() # moveDocument() # markAsProcessFile() Extends Extends GroceryFileProcess DrugstoreFileProcess # validateName() # validateName() # processFile() # processFile() # createLog() # createLog() OnMemoryDataBase Log - PROCESS DOCUMENTS: MutableMap - CUSTOMERS = intArrayOf() + buildLog(id: Int, customer: Int, date:String,amount: Double): String + getFileStatus(fileName: + create(fileName: String, data: String) String): String + setProcessFile(fileName: String) + customerExist(id: Int): Boolean

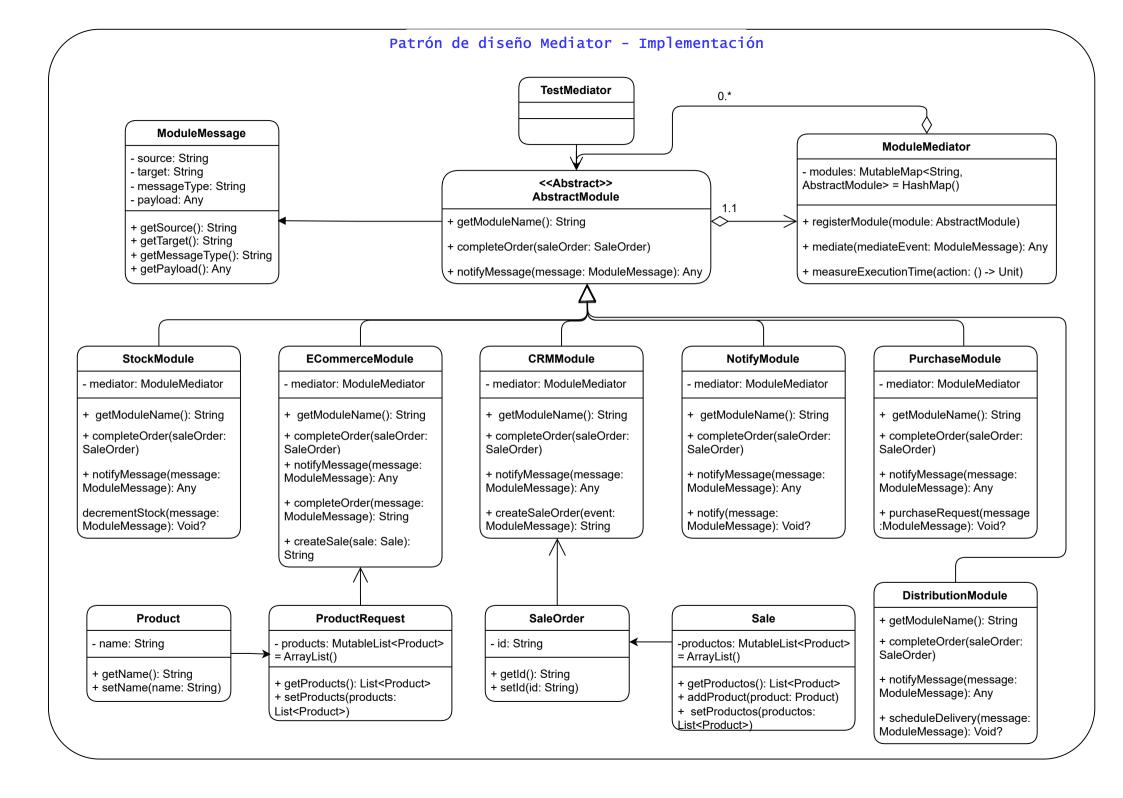


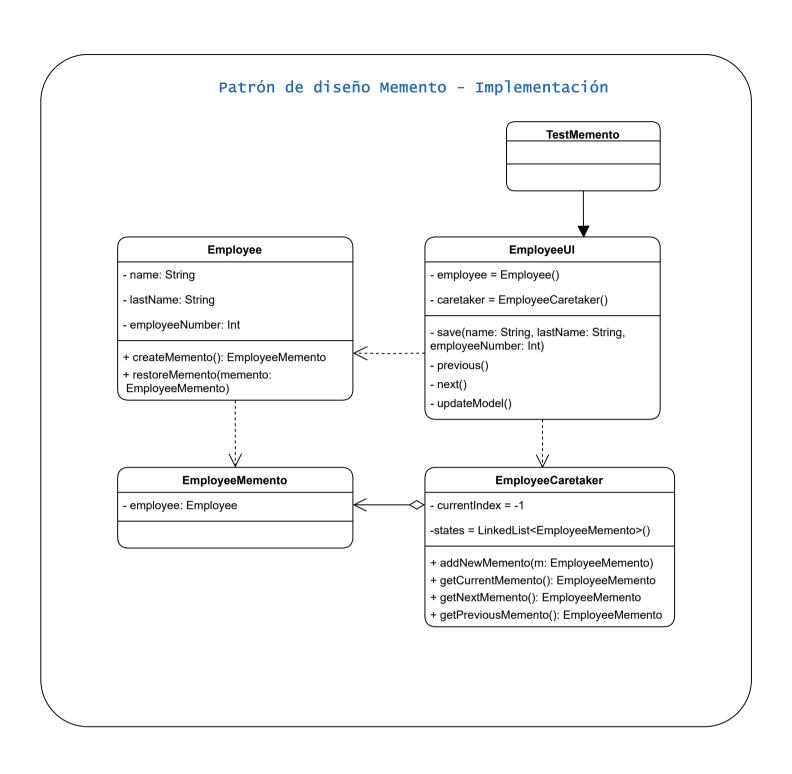
#### Patrón de diseño Chain Of Responsability - Implementación Estructura de la orden CreditData Telephone **Address** - creditLimit: Double - lada: String - address1: String **Status** - balance: Double - number:String - address2: String - ACTIVO - ext: String - country: String - BAJA - cp: String Contributor Customer - name: String, - name: String, - rfc: String, - rfc: String, - status: Status, - status: Status, -Extends-- address: Address, - address: Address, - telephone: Telephone, - telephone: Telephone, - creditData: CreditData, - creditData: CreditData <<Abstract>> OrderItem Order - product: Product, - createDate: LocalDate - price: Double, - contributor: Contributor - quantity: Double - orderItems: List<OrderItem> + getTotal(): Double getTotal(): Double Extends SalesOrder **Product** + Order() - name: String listPrice: Double

### Patrón de diseño Chain Of Responsability - Implementación OrderValidatorBuilder **TestChainOfResponsability** <<Abstract>> OrderValidator -buildSalesOrderValidator(): -Build Chaln-→ OrderValidator -validators: MutableList<OrderValidator> = ArrayList() -buildCustomerValidator(): OrderValidator + validate(order: Order) + addValidator(validator: OrderValidator) -buildContributorValidator(): OrderValidator Extends ContributorValidator SalesOrderValidator **TelephoneValidator** AddressValidator + validate(order: Order) + validate(order: Order) + validate(order: Order) + validate(order: Order) CreditValidator CustomerValidator OrderItemValidator ValidationException + validate(order: Order) + validate(order: Order) + validate(order: Order) - message: String

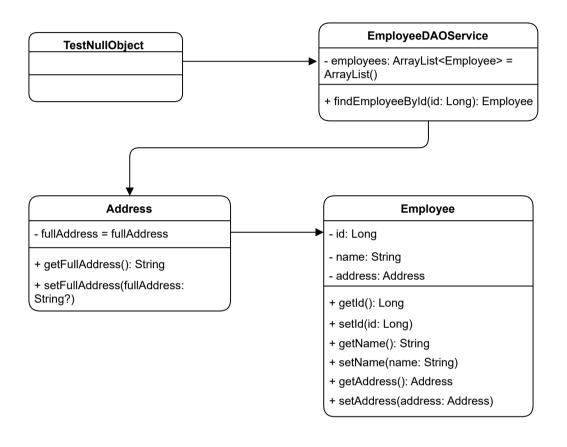








## Patrón de diseño NullObject - Implementación



#### Patrón de diseño State - Implementación **TestState** ServerPanel MessageProcess - server: Server - messageCounter = 0 sendThread: Thread? = null - server: Server + sendMessageEvent(evt: ActionEvent) - messageQueue:Queue<String> = LinkedList() + startAction(evt: ActionEvent) + getState(): AbstractServerState + queueMessage(message: String): Boolean + stopMessageProcess() + countMessage(): Int + start() Server + stop() - messageProcess: MessageProcess - state: AbstractServerState + setState(state: AbstractServerState?) <<Abstract>> + handleMessage(message: String) AbstractServerState + getState(): AbstractServerState handleMessage(server: Server?, + stopMessageProcess() message:String?) SaturatedServerState **StartServerState** StopServerState StartingServerState - monitoringThread: Thread - monitoringThread: Thread - monitoringThread: Thread - monitoringThread: Thread + handleMessage(server: + handleMessage(server: + handleMessage(server: + handleMessage(server: Server?, message: String?) Server?, message: String?) Server?, message: String?) Server?, message: String?)

