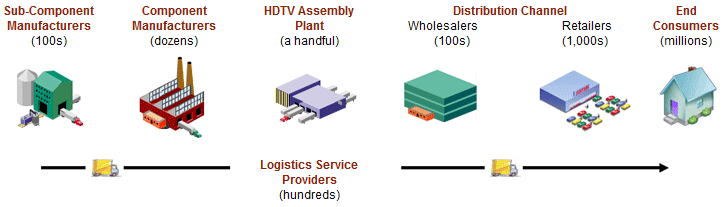
**Supply Chain Pattern**



**Client (Consumer)** – “*orders”*

* *Client calls orderSomeService(data), which calls a manager.*

|

|\_\_ **Manager (Retailer)** – “*delegates”*

* *delegateSomeAction(data), which calls the controllers in order of need.*

|

|\_\_ **Controller (Wholesaler, Contractor)** – “*dispatches”*

* *dispatchSomeAction(data), which calls the appropriate classes and methods to do the work.*

|

|\_\_ **Builder (Assembly Plant)** – “*constructs”*

* *contructMutatedObject(data), which accepts data and calls factories to construct the object.*

|

|\_\_ **Factory (Assembly Plant)** – “*assembles”*

* *assembleObject(), which creates an instance of an object.*

|

|\_\_ Interface (Component Manufacturer)

|

|\_\_ Implementation (Sub Component Manufacturer)

**Tree Structure:**

Client

application client sends an order to a manager

public Client

public Client()

public clientMethod()

private orderClientMethods()

\/

Manager

request manager delegates the orders to controllers

public ManagerOne exends ControllerOne

public ManagerOne()

private Object dtoObject

public managerOneMethods()

// --- private delegate methods ---

private delegateManagerOneMethods()

/ | \

Controller <-> Controller <-> Controller

each controller is a grouping of related functionality and dispatches object construction to builders

public ControllerOne extends BuilderOne public ControllerTwo public ControllerThree extends BuilderThree

protected ControllerOne() protected ControllerTwo() protected ControllerThree()

protected controllerOneMethods() protected controllerTwoMethods() protected controllerTwoMethods()

// --- private dispatch methods --- // --- private dispatch methods --- // --- private dispatch methods –

private dispatchControllerOneMethods() private dispatchControllerTwoMethods() private dispatchControllerThreeMethods()

| | |

Builder <-> Builder <-> Builder

each builder is a grouping of related functionality that construct object state from factory objects

public BuilderOne extends FactoryOne public BuilderTwo public BuilderThree extends FactoryThree

protected builderMethods() protected builderTwoMethods() protected builderTwoMethods()

// --- private construct methods --- // --- private construct methods --- // --- private construct methods ---

private construct BuilderOneMethods() private construct BuilderTwoMethods() private construct BuilderOneMethods()

| | |

Factory <-> Factory <-> Factory

each factory is a grouping of related behaviors that assemble objects

public FactoryOne public FactoryTwo public FactoryThree

public FactoryOne() public FactoryTwo() public FactoryThree()

protected factoryMethods() protected factoryTwoMethods() protected factoryTwoMethods()

// --- private assemble methods --- // --- private assemble methods --- // --- private assemble methods ---

private assembleFactoryOneMethods() private assembleFactoryTwoMethods() private assembleFactoryOneMethods()

**Possible Module Structures:**

1. Multi-Module Project

apis - interfaces( boundaries )

models - entities, beans( builders, entities, factories )

resources - method annotations( resource )

services - business logic( managers, controllers, services, daos )

utilities - common functionality, configurations( listeners, parameters ,helpers)

web - deployment, view( WEB-INF, META-INF )

**Possible Package Structures:**

1)

collaborators.boundaries

collaborators.builders

collaborators.entities

collaborators.factories

implementations.controllers

implementations.daos

implementations.delegates

implementations.managers

implementations.resources

implementations.services

utilities.constants

utilities.helpers

utilities.listeners

utilities.parameters