**Utilizando defmulti...**

(def task1 {:type "A" :data "Task A Data"})

(def task2 {:type "B" :data "Task B Data"})

(def task3 {:type "C" :data "Task C Data"})

(defn execute-task-A [task] (println "Executing task [" (:type task) "] with data [" (:data task) "]"))

(defn execute-task-B [task] (println "Executing task [" (:type task) "] with data [" (:data task) "]"))

(defn execute-task-C [task] (println "Executing task [" (:type task) "] with data [" (:data task) "]"))

(defmulti **execute-task** (fn [task] (:type task)))

(defmethod execute-task "A" [task] (execute-task-A task))

(defmethod execute-task "B" [task] (execute-task-B task))

(defmethod execute-task "C" [task] (execute-task-C task))

(execute-task task1)

(execute-task task2)

(execute-task task3)

**Utilizando composición de funciones...**

(defn **execute** [command & args]

(apply command args))

(execute execute-task-A task2)