**UNIVERSIDAD UTE**

**FACULTAD DE CIENCIAS DE LA INGENIERÍA E INDUSTRIAS**

**APLICACIONES DISTRIBUIDAS**

**TEMA: INVERSIÓN DE CONTROL**

**ELABORADO POR:**

**STEVE CABEZAS**

**ANDREA LLIVE**

**JAVIER SANCHEZ**

**OCT 2020 – FEB 2021**

# 

# **1. Inversión de control.**

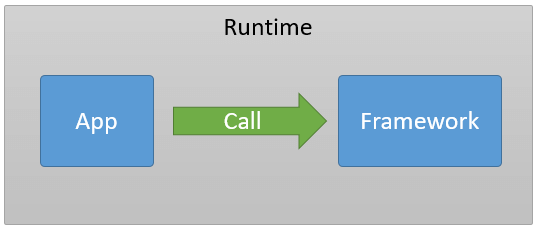
Es un principio de diseño de software en el que el flujo de ejecución de un programa se invierte respecto a los métodos de programación tradicionales.

la inversión de control se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que algún tipo de entidad o arquitectura externa lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir.

Esta nueva filosofía es muy útil cuando se usan frameworks de desarrollo. Es el framework el que toma el control, el que define el flujo de actuación o el ciclo de vida de una petición. Es decir, es el framework quien ejecuta el código de usuario

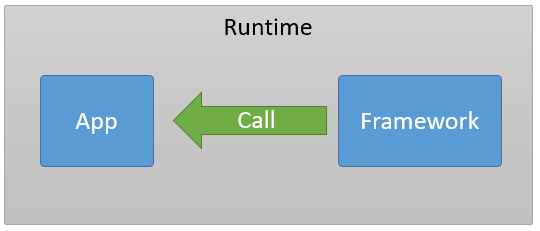
**Funcionamiento.**

Inversion of control (IoC) es una forma de trabajar que rompe con el formato tradicional, en donde el programar es el encargado de definir la secuencia de operaciones que se deben de realizar para llegar a un resultado. En este sentido el programador debe conocer todos los detalles del framework y las operaciones a realizar en él, por otra parte, el Framework no conoce absolutamente nada de nuestro programa. Es decir, solo expone operaciones para ser ejecutada. La siguiente imagen muestra la forma de trabajo normal.



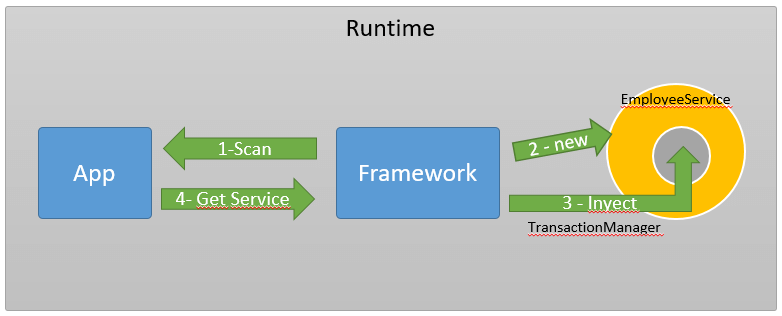
Veamos que nuestra aplicación es la que llama en todo momento al Framework y este hace lo que la App le solicita. Por otra parte, al Framework no le interesa en absoluto la App.

Pero qué pasaría si invirtiéramos la forma de trabajo, y que en lugar de que la App llame funciones del Framework, sea el Framework el que ejecute operaciones sobre la App, ¿suena extraño no? Pues en realidad a esta técnica es a la que se le conoce como Inversion of Control. Veamos en la siguiente imagen como es que funciona IoC:



Esta última imagen se ve bastante parecida a la anterior, sin embargo, tiene una gran diferencia, y es que las llamadas se invirtieron, ahora es el Framework es el realiza operaciones sobre nuestra App.

Seguramente a estas alturas te pregunta cómo es que un Framework que no conoce tu App y que incluso se compilo desde mucho antes puede realizar operaciones sobre nuestra App. La respuesta es simple pero complicada, y es que utilizar este IoC es mucho más simple de lo que parecería, pero implementar un Framework de este tipo tiene un grado alto de complejidad, dicho de otra manera, si tú eres la App puedes implementar muy fácilmente al Framework, pero si tu estas desarrollando el Framework puede ser un verdadero dolor de cabeza.



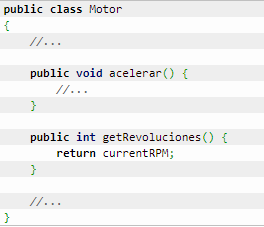
Analicemos la imagen, primero el Framework analizará al App mediante los metadatos, este encontrará la clase EmployeeService que esta anotada con @Service, seguido le inyectará una instancia del TrasnactionManager mediante el método setTransactionManager que esta anotado con @Resource, finalmente el servicio será retornado a la App para ser utilizado.

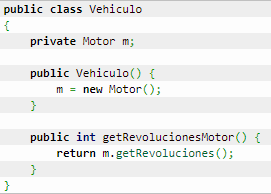
Cabe mencionar que los proxys son clave para controlar como la forma en que el usuario utiliza las operaciones. Puedes encontrar mucho más de este patrón en mi libro.

**Inversión de control mediante la inyección de dependencias**

La dependencia de un objeto respecto a otro ocurre cuando el primero necesita del segundo para completar alguna de sus tareas.

Un caso típico es el de un coche, que depende del motor para, por ejemplo, poder avanzar.

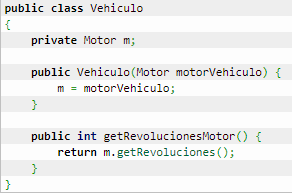




Como podemos comprobar, la dependencia entre las clases Vehiculo y Motor queda patente dado que una instancia de la primera alberga dentro una instancia de la segunda.

Sin embargo, podemos comprobar que el acoplamiento existente en el código es alto. El motor está fuertemente ligado al vehículo, de forma que esta relación es poco flexible. Si quisiéramos realizar cualquier tipo de modificación en la clase Motor, esto supondría un alto impacto en la clase Vehiculo (por ejemplo, si quisiéramos hacer una concreción en MotorDiesel o MotorGasolina).

Como primer paso para desacoplar el motor del vehículo, podríamos hacer que la clase Vehiculo deje de encargarse de instanciar el objeto Motor, pasándoselo como parámetro al constructor. De esta forma, la clase Vehiculo quedaría de la siguiente manera:



El constructor de vehículo se encarga de inyectar la dependencia dentro del objeto, eliminando esta responsabilidad de la propia clase. De esa forma, hemos dado un paso para desacoplar ambos objetos.

El siguiente paso que podríamos dar en aras de continuar con el desacoplamiento de ambos objetos es el uso de interfaces.

**Diferentes formas de inyección de dependencia**

Como hemos visto en el ejemplo anterior, una de las formas de realizar la inyección de dependencias es mediante el uso de un constructor, pero no es la única. De hecho, a nivel formal podemos decir que existen algunas más:

* Inyección por constructor
* Inyección por método
* Inyección por propiedad (esta la encontramos en otros lenguajes de programación, como C#).

**Ventajas**

* Proporciona modularidad.
* Permite ampliar la funcionalidad de nuestras aplicaciones sin modificar las clases.
* Evita la dependencia entre las clases.
* Flexibiliza nuestras aplicaciones haciéndolas mas configurables.

# **2. Autoevaluación.**

|  |  |  |
| --- | --- | --- |
| **INTEGRANTES** | **NOTA** | **OBSERVACIONES** |
| STEVE CABEZAS | 100% | Tuvimos una buena comunicación en el grupo y así logramos una buena exposición. |
| ANDREA LLIVE | 100 % | Se Trabajo equitativamente dentro del grupo, sin ninguna novedad. |
| JAVIER SANCHEZ | 100 % | No hubo inconvenientes al realizar el presente trabajo, todo se dividió de manera equitativa. |

# **3. Referencias.**

➔ Blancarte Oscar,El concepto Inversion of Control (2020) recopilado 23 de Noviembre de 2020 https://www.oscarblancarteblog.com/2016/12/01/concepto-inversion-of-control/

➔https://www.youtube.com/watch?v=gGkeH38XMLk&ab\_channel=ManuelZapata

➔https://www.youtube.com/watch?v=Cs1HN6pEg4&ab\_channel=pildorasinformaticas