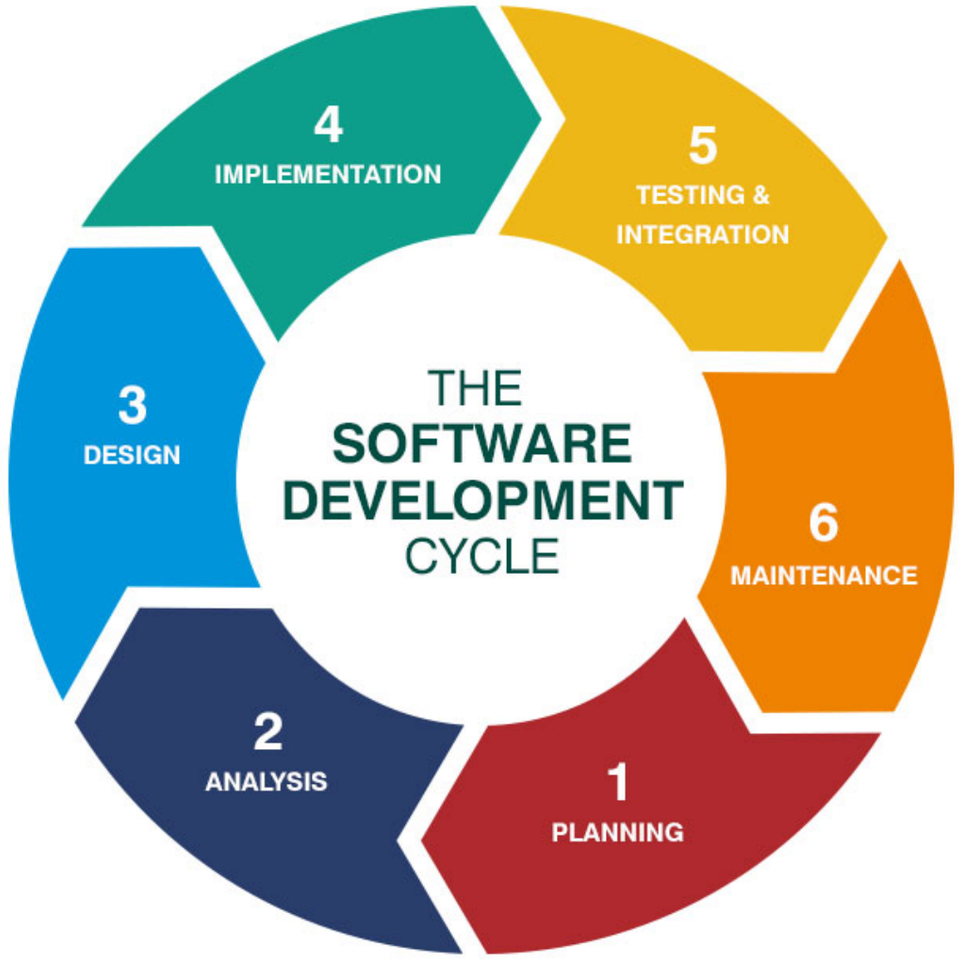
Programación de Aplicaciones Telemáticas

TEMA 1: LA WEB EN LA INDUSTRIA

AGENDA

- Ciclo de vida del software
- Paradigmas de programación
- Sistemas distribuidos
- Ecosistema Frontend
- Ecosistema Backend
- Arquitecturas de software
- Tendencias

CICLO DE VIDA DEL SOFTWARE



PARADIGMAS DE PROGRAMACIÓN

- Estructurada
- Imperativa
- Orientada a Objetos
- Declarativa
- Funcional
- Reactiva

PARADIGMAS DE PROGRAMACIÓN

La programación estructurada es un paradigma de programación orientado a mejorar la claridad, calidad y tiempo de desarrollo de un programa de computadora recurriendo únicamente a subrutinas y tres estructuras básicas: secuencia, selección (if y switch) e iteración (bucles for y while); asimismo, se considera innecesario y contraproducente el uso de la instrucción de transferencia incondicional (GOTO)

PARADIGMAS DE PROGRAMACIÓN

La programación imperativa (del latín imperare = ordenar) es el paradigma de programación más antiguo. De acuerdo con este paradigma, un programa consiste en una secuencia claramente definida de instrucciones para un ordenador.

PARADIGMAS DE PROGRAMACIÓN

```
int sum = 0;

for (int counter = 1; counter < limit; counter++) {
        if ((counter % THREE == 0) || (counter % FIVE == 0)) {
                sum += counter;
        }
}

return sum;
```

PARADIGMAS DE PROGRAMACIÓN

La programacion Declarativa: que está basado en el desarrollo de programas especificando o "declarando" un conjunto de condiciones, proposiciones, afirmaciones, restricciones, ecuaciones o transformaciones que describen el problema y detallan su solución.

PARADIGMAS DE PROGRAMACIÓN

La Programación Orientada a Objetos (POO, en español; OOP, según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos se utilizan como metáfora para emular las entidades reales del negocio a modelar.

PARADIGMAS DE PROGRAMACIÓN

La programación funcional es un paradigma de programación declarativa basado en el uso de verdaderas funciones matemáticas. En este estilo de programación las funciones son ciudadanas de primera clase, porque sus expresiones pueden ser asignadas a variables como se haría con cualquier otro valor; además de que pueden crearse funciones de orden superior

PARADIGMAS DE PROGRAMACIÓN

```
BiPredicate<integer, integer=""> isMultiple = (l, i) -> l % i
Predicate<integer> isMultiple3 = number -> isMultiple.test(num
Predicate<integer> isMultiple5 = number -> isMultiple.test(num

return IntStream.range(1, limit).boxed()
                .filter(isMultiple3.or(isMultiple5))
                .reduce(0, Integer::sum);

                                                    </integer></in
```

Programación reactiva es un paradigma basado en la declaración de una serie de objetos emisores de eventos asíncronos y otra serie de objetos que se "suscriben" a los primeros (es decir, quedan a la escucha de la emisión de eventos de estos) y *reaccionan* a los valores que reciben.

PARADIGMAS DE PROGRAMACIÓN

```
Observable< Integer> threes = Observable.range(1, 999)
        .map(e -> e * 3).takeWhile(e -> e < 1000);
Observable< Integer> fives = Observable.range(1, 999)
        .map(e -> e * 5).takeWhile(e -> e < 1000);
Observable< Integer> threesAndFives = Observable
        .merge(threes, fives).distinct();
Single< Integer> summer = threesAndFives
        .reduce(0, (a, b) -> a + b);
summer.subscribe(System.out::print);
```

SISTEMAS DISTRIBUIDOS

Un sistema distribuido es un sistema cuyos componentes están ubicados en diferentes computadoras en red, que se comunican y coordinan sus acciones pasándose mensajes entre sí. Los componentes interactúan entre sí para lograr un objetivo común.

Source: Wikipedia

ECOSISTEMA FRONTEND

En el desarrollo de software, el Frontend es la parte del software que interactúa con los usuarios a través del navegador web y las tecnologias web como son HTML, CSS y Javascript.

Interface de usuario web, recibe los datos del usuario y los enviará al Backend para su procesamiento.

ECOSISTEMA FRONTEND

El desarrollo en Frontend tiene que tener en cuenta los siguientes conceptos:

- Interface de usuario
- Accesibilidad
- Look & Feel
- Interacción

ECOSISTEMA BACKEND

El Backend se encarga de recibir la información del usuario y procesarla. Pueden existir los siguientes modelos de interacción:

- request/response
- fire-and-forget
- request/stream
- channel

ECOSISTEMA BACKEND

En el desarrollo del Backend, se desarrollan endpoints para procesar la informacion y se interacciona con distintos elementos como:

- REST APIs
- XML Web Services
- Bases de datos
- Caches
- Brokers de mensajeria

ECOSISTEMA BACKEND

No todo el desarrollo de Backend se desarrolla en Java en el lado del Backend o en Javascript en el lado del cliente. Existen más opciones:

- Scala
- Kotlin
- Go
- Rust
- Typescript

ARQUITECTURAS DE SOFTWARE

Dependiendo del escenario de uso del Backend, podrias elegir entre:

- Desarrollo síncrono REST
- Desarrollo asíncrono REST
- Desarrollo event-driven
- Serverless

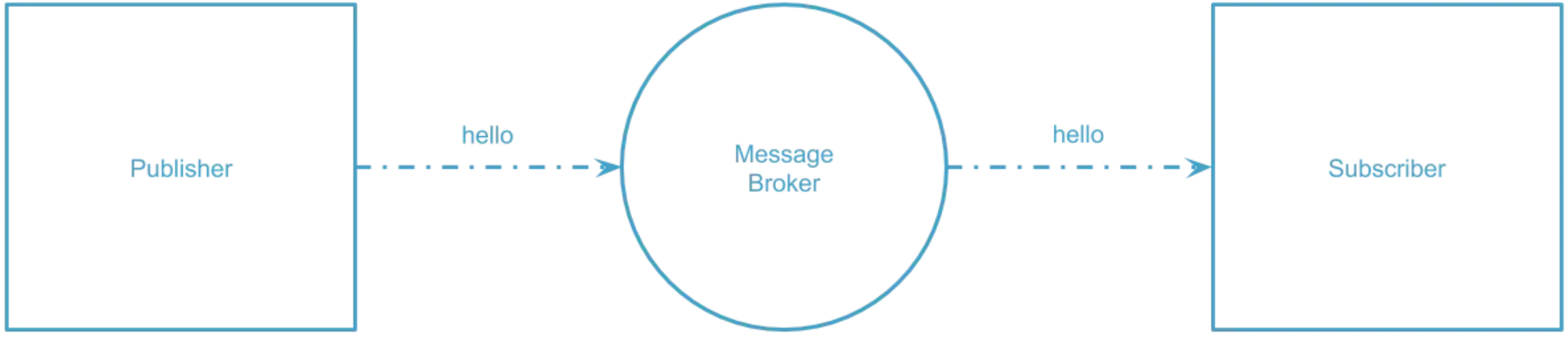
ARQUITECTURAS REST

REST es una arquitectura de desarrollo web que puede ser utilizada en cualquier cliente HTTP.

- Es un protocolo sin estado
- Las operaciones se definen a traves de los metodos: GET, POST, PUT, PATCH y DELETE
- Utiliza URIs
- Emplea hipermedia (HTML, XML & JSON)

ARQUITECTUAS EVENT-DRIVEN

La Event-Driven Architecture (EDA, por sus siglas en inglés) es una arquitectura en la que el software ejecuta una acción al recibir una o más notificaciones de eventos.



ARQUITECTUAS EVENT-DRIVEN

COMPONENTES:

- Message broker
- Publisher/Subscriber
- Message
- Channels

ARQUITECTURAS SERVERLESS

Las Arquitecturas Serverless se caracterizan por la capacidad de computación efímera en infraestructura manejada por proveedores cloud tipo AWS, Azure o Google Cloud.

TENDENCIAS