**UNIVERSIDAD DE TARAPACÁ**



**FACULTAD DE INGENIERÍA**

Departamento de Ingeniería en Computación e Informática



**Ensayo: Estilos de arquitectura**

**Autor**: Iván Cardemil

Patricio Tudela

**Curso**: Sistemas Distribuidos

**Profesor**: Diego Aracena

**Arica, 09 de septiembre de 2019.**

**Arquitectura de software**

La arquitectura de software es la organización lógica de un sistema distribuido dentro de componentes de software.

Se está de acuerdo en la definición de arquitectura de software ya que sienta las bases en cómo se puede resolver un problema en la informática entregando formas y guías para ello. Según las palabras de David Garlan y Mary Shaw, “más allá de los algoritmos y estructuras de datos de la computación”. De esto año tras año se fueron ideando y descubriendo nuevas formas de diseñar arquitecturas de sistemas para cumplir con requisitos, que con otros no pueden ser cumplidos o son más eficientes.

**Estilo de arquitecturas**

Los estilos de arquitecturas son formulados en base a “componentes”, las forma en que estos están conectados entre sí, los datos intercambiados entre estos y finalmente como todos estos elementos son unidos en un sistema.

Un **componente** es una unidad modular proveídas con interfaces bien definidas, que pueden ser reemplazadas en sus ambientes. Es importante en muchos casos que, al retirar un componente de un sistema, o que, a lo más, ciertas partes dejen de funcionar temporalmente. Entonces reemplazar un componente solo se puede hacer cuando sus interfaces permanecen inalteradas.

Otro concepto es el de **conector** el cual es un mecanismo que intermedia entre las comunicaciones, la coordinación o cooperación entre componentes.

Usar estos componentes y conectores permiten realizar varias configuraciones en un sistema lo que, a día de hoy, son clasificadas como estilos de arquitectura. Algunos de los estilos de arquitectura en los sistemas distribuidos son:

**Arquitectura basada en niveles**

En estas arquitecturas los componentes están organizados por niveles, en donde un componente superior realiza una llamada a un componente inferior, que luego generalmente espera una respuesta. En algunos casos un componente inferior realiza una llamada a uno superior.

Para estas arquitecturas también existen los protocolos de arquitectura basadas en niveles donde unos de los más conocidos son los “protocolos de comunicación en stacks”. En estos protocolos cada nivel implementa uno o más servicios de comunicación pudiendo enviar datos desde un destino a muchos objetivos. Cada nivel ofrece una interfaz especificando sus funciones que pueden ser llamadas. Otro concepto en el caso de la comunicación es el de “protocolos de comunicación” donde se describen las reglas que las “partes” seguirán para intercambiar información. Es importante diferenciar el servicio que ofrece un nivel de la interfaz por el cual un servicio es hecho disponible, y el protocolo que un nivel implementa para establecer la comunicación. Por lo tanto, para arquitecturas con 2, 3 o más niveles es necesario establecer protocolos para la comunicación entre estos, con diferentes servicios.

**Arquitecturas basadas en objetos**

Algunas organizaciones siguen arquitecturas basadas en objetos. Cada uno de estos objetos representa a un componente, y estos están conectados por procedimiento de llamadas. En los sistemas distribuidos los procedimientos pueden también tomar parte en una red, donde el objeto llamado no necesariamente debe estar en la misma máquina que el objeto que lo ha llamado.

Las arquitecturas basadas en objetos proveen formas naturales de encapsular datos, lo cual es llamado el “estado del objeto”, y las operaciones que este puede realizar sobre esos datos son llamados “métodos del objeto”. La interfaz ofrecida por un objeto implementación y detalles permitiendo considerar a un objeto independiente del ambiente. Esto permite reemplazar un objeto por otro teniendo exactamente la misma interfaz. Esta implementación de objetos nos permite colocar las interfaces en una máquina mientras que los objetos en sí se encuentran en otra.

# **Arquitectura centrada en recursos**

Debido al aumento de servicios disponibles sobre la web, la arquitectura de los servicios de sistemas distribuidos ha tenido que ser repensado, ya que se enfrenta a un desafío muy importante al momento de integrar varios componentes.

Es por eso, que, como alternativa, se da otro enfoque a los sistemas distribuidos, y se pasan a ver como una gran colección de recursos que pueden ser manejados de forma individual por los componentes, donde estos recursos pueden ser añadidos, removidos, leídos y modificados por aplicaciones remotas. Este estilo de arquitectura, que es muy utilizado en la web, es conocido como el estilo de Transferencia de Estado Representacional (**REST**) y quienes lo adoptan deben seguir los siguientes principios:

* Los recursos deben ser identificados por un único nombre de su esquema.
* Todos los servicios deben ofrecer la misma interfaz de cuatro operaciones.
* Los mensajes desde o hacia un servicio deben ser descritos completamente.
* Después de ejecutar la operación del servicio, el componente olvida todo acerca del cliente que lo llamó. A esta propiedad se le llama **stateless**.

En las arquitecturas que aplican el diseño RESTful se pueden identificar las siguientes operaciones:

* PUT: Crea un nuevo recurso
* GET: Obtiene el estado de un recurso
* DELETE: Elimina un recurso
* POST: modifica el estado de un recurso

Estas operaciones se realizan a través de peticiones utilizando el protocolo HTTP, donde se debe especificar la el identificador del recurso donde se va a operar.

Es patrón se ha vuelto muy popular debido a su simplicidad. Pero con el paso de los debates, algunos han planteado las desventajas de la arquitectura RESTful, ya que pueden entorpecer el uso de soluciones sencillas en la comunicación de esquemas, por ejemplo, si es que un sistema desea hacer seguimiento al estado de una ejecución, y como se vio en la descripción de los principios, el componente olvida todo lo relacionado al cliente que lo llamó.

# **Arquitectura basada en eventos (publicador/suscriptor)**

Una gran cantidad de sistemas distribuidos han adoptado la arquitectura que separa fuertemente los procesos y la coordinación.

La combinación de sistemas desacoplados por referencia y acoplados temporalmente forman el grupo de modelos para la coordinación basada en eventos. Donde la única cosa que lo procesos pueden hacer es la de publicar una notificación describiendo la ocurrencia de un evento. En estos casos el proceso debe suscribirse a un tipo específico de notificación. En el caso ideal de estos sistemas, es esperable que, al momento de publicarse una notificación, los procesos deben entregar esta información a quienes se han suscrito.

La clave de la comunicación entre los procesos, es la de intercambiar información en el espacio de compartición de información utilizando tuplas, donde estas, se pueden obtener por medio de patrones de búsqueda.

La forma de operar en el la coordinación de eventos es la siguiente:

* Un proceso provee de tuplas utilizando un patrón de búsqueda.
* Otro proceso inserta una tupla en el espacio de información que satisface al patrón de búsqueda.
* Los suscriptores son notificados.

A lo descrito con anterioridad se le llama arquitectura publicador/suscriptor. Donde sus desafíos están en la eficiencia y escalabilidad de la implementación de los sistemas de emparejamiento de suscripciones para notificar a los usuarios.