# Vortex 2.0 – Mensajes y respuestas

La idea de este documento es especificar y bajar un poco las ideas que estuvieron dando vueltas a la hora de definir la nueva versión de vortex. Sin entrar en demasiado detalle el objetivo es que se entienda como interactuar con un servidor vortex como cliente.

## Componentes de un mensaje según Vortex

El componente fundamental de vortex es el mensaje. El MensajeVortex en el protocolo 2.0 está compuesto de tres partes:

* Identificación: Es utilizada por la red para detectar mensajes duplicados. Está conformada de manera tal que dos mensajes distintos tengan muy poca probabilidad de tener la misma
* Contenido: Es el contenido que tiene valor para el emisor y receptor. Está descompuesto en dos partes. El valor específicamente, y el tipo de contenido que es una ayuda para la red
* tagsDestino: Es un conjunto de Tags aplicados al mensaje que permite a la red identificar los clientes interesados en el mensaje

## Servidor HTTP

El único servidor que está funcionando públicamente actualmente sólo utiliza HTTP para enviar y recibir mensajes. Por lo que cada mensaje vortex, debe ser encapsulado en un objeto contenedor que tiene datos sobre la sesión http.

* sessionId: Es un id que otorga el servidor cuando un cliente abre sesión para poder identificarlo del resto
* mensajes: Es el conjunto de mensajes que el cliente envía o recibe

Todos los mensajes al servidor son enviados dentro de este contenedor, y son recibidos de la misma manera

## Envío de mensaje (sin recepción)

La forma más básica de vortex sólo implica el envío de mensajes por parte de un cliente, sin identificarse. Esta forma está pensada para sensores muy simples (que no requieren feedback). Esto posibilita la implementación de vortex en hardware sin necesidad de controladores complejos.

### Request HTTP para vortex

Para que un mensaje vortex llegue al servidor es necesario enviar un request POST a la dirección:  
<http://kfgodel.info/vortex/controllers/main>

El request sólo debe contener un parámetro POST con el nombre “mensajeVortex”. El valor del parámetro será un WrapperVortex convertido a JSON.

Ejemplo:

## Ejemplos de mensajes y sus respuestas

#### Hola mundo a la Vortex

Enviado:

{

"sessionId":null,

"mensajes":[

{

"identificacion":

{

"hashDelContenido":"1",

"idDelEmisor":"1",

"numeroDeSecuencia":1,

"timestamp":0

},

"contenido":

{

"tipoContenido":"Hola mundo",

"valor":"texto"

},

"tagsDestino":[

"TagEjemplo"

]

}

]

}

Recibido:

{

"sessionId":null,

"mensajes":[]

}

Este tipo de mensajes tiene respuesta vacía. El nodo vortex hará lo posible por rutearlo pero el emisor no podrá conocer el éxito o fracaso del ruteo. El nodo no tiene forma de identificar al emisor.

#### Declaración de Tags

Enviado: {

"sessionId":null,

"mensajes":[

{

"identificacion":

{

"hashDelContenido":"1",

"idDelEmisor":"1",

"numeroDeSecuencia":2,

"timestamp":0

},

"contenido":

{

"tipoContenido":"vortex.metamensaje.ReemplazarTags",

"valor":"{\"tags\":[\"TAG1\"]}"},

"tagsDestino":[

"CHE"

]

}

]

}

Recibido:

{

"sessionId":1,

"mensajes":[]

}

Todos los metamensajes generan una sesión si no existía, por eso el id de sesión. Este tipo de mensajes cambia el estado del cliente en el servidor, de manera que los mensajes que circulen por el servidor sean ruteados hacia el cliente, si coinciden con el tag declarado “TAG1” en este caso. El tipo de contenido debe ser respetado